الفصل الأول

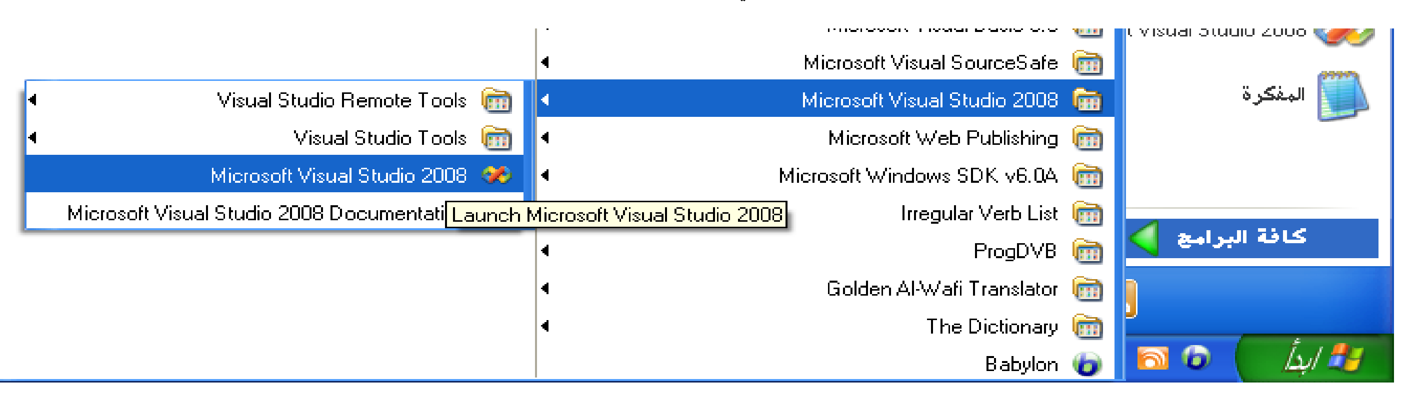
البداية مع فيجوال أستوديو 2008

مقدمة: ما هي فيجوال بيسك 2008 ؟
فيجوال بيسك2008 هي اداة تطوير يمكنك استخدامها لبناء تطبيقات برمجية software applications تنجز عمل مفيد ليبدو عظيما من خلال اعداداته المتنوعة .باستخدام فيجوال بيسك2008 تستطيع ان تنشئ تطبيقات لنظام التشغيل ويندوز,تطبيقات الانترنت, تطبيقات النقال(الهاتف المحمول),
الفائدة الاكثر اهمية للفيجوال بيسك هي انها تم تصميمها بحيث تعمل على زيادة قدرتك الانتاجية في عملك البرمجي اليومي .,وخصوصا اذا كنت تريد استخدام المعلومات في قواعد البيانات أو انشاء حلول للإنترنت.ولكن الفائدة الهامة هي حالما تعتاد على الادوات في الفيجوال أستودو2008 والتي الفيجوال بيسك 2008 هي احدى اللغات المدعومة من قبلها,تستطيع استخدام هذه الادوات لكتابة البرامج للفيجوال2008 C++ والفيجوال سي شارب2008 C#, والويب Web Developer 2008 ومكونات اخرى.....
تاتي الفيجوال استوديو2008 في عدة طبعات(اصدارات مختلفة لنفس المنتج) متضمنة الطبعة القياسية standard edition, الطبعة الاحترافية Professional Edition, والطبعة المعدة لعمل فريق Team Suite وأخيرا الطبعة السريعة Express Edition(مايناسب هذا الكتاب هو الطبعة القياسية على الرغم من ان بعض الادوات ستكون غير موجودة فيها مثل ادوات قواعد البيانات لذلك من الافضل استخدام الطبعة الاحترافية) ويوجد ايضا بعض الاختلاف لمن استخدام اصدار سابق من الفيجوال استوديو نت كالفيجوال استوديو 2005 وهذه الاختلافات هامة, لذلك اوصيك باستخدام فيجوال استوديو 2008 وبشكل خاص الطبعة الاحترافية.
نظام التشغيل المطلوب من اجل الفيجوال استوديو 2008
ستحتاج الى التجهيزات والبرمجيات التالية لاتمام التدريب في هذا الكتاب:
- ✓ نظام التشغيل ويندوز فيستا أو اكس بي سيرفس باك 2 أو ويندوز سيرفر 2003 سيرفس باك 1
- ✓ ميكروسوفت فيجوال استوديو 2008 (إما الطبعة القياسيةStandard Edition, أو الطبعة الاحترافية Professional Edition، اوالطبعة المعدة لفريق عمل Team Suite)
- ✓ تجهيزات الكمبيوترالأدنى المطلوبة : معالج GHz 1.6 غيغا هرتز, 384 MB RAM ميغابايت من الذاكرة العشوائية,1024×768 شاشة عرض, القرص الصلب 5400 RPM دورة/دقيقة, .
- ✓ تجهيزات الكمبيوتر الموصى بها: 2.2 غيغا هرتز أو اعلى 1024 ميغابايت من الذاكرة العشوائية أو اعلى شاشة 1280×1024 (من اجل ويندوز فيستا :معالج لايقل عن 2.4 GHz و768 MB RAM (الذاكرة العشوائية) وباقي التجهيزات الضرورية المتوفرة عند الجميع على ما أظن.

سأفترض انك نصبت واحد من إصدارات الفيجوال أستوديو 2008 من اجل هذا الكتاب فاني سأستخدم طبعت فيجوال أستوديو الاحترافية professional edition ولكن من اجل كل شيء تم مناقشته في هذا الكتاب يمكن أن يطبق أيضا باستخدام الطبعة القياسية standard edition ولكن بعض ميزات الطبعة الاحترافية والتي لا تكون مدعومة بالطبعة القياسية فيما يخص أدوات قواعد البيانات التي شرحت في الفصل من 21 إلى الفصل 24 , يبدأ هذا الكتاب بخلاصة عن الفيجوال أستوديو وأدواتها الأساسية ولا يتطلب منك معرفة مسبقة في VB6, فقط بعض المعرفة بالبرمجة بشكل عام والفيجوال بيسك هي واحدة من لغات البرمجة ضمن الفيجوال أستوديو التي تستطيع بناء تطبيقاتك بها , وقد اتبعت الإقناع والبساطة في هذا الكتاب وما يجب أن تتذكره دائما هوان فيجوال أستوديو هي بيئة تطوير متكاملة لبناء واختبار وتشغيل وتصحيح الأخطاء وتوزيع تطبيقات متنوعة مثل تطبيقات ويندوز windows applications وتطبيقات الويب web application وأدوات التخصيص والفئات classes and custom controls وحتى تطبيقات console وبالتالي فإنها تدعم عدد ضخم من الأدوات المرئية visual tools لانجاز العديد من التصميم المشترك ومهمات البرمجة وميزات أخرى إضافية أكثر من أن يستطيع أي مؤلف تغطيتها. في هذا الفصل سوف تتعلم ما يلي:
1 - استكشاف بيئة التطوير المتكاملة integrated development environment of Visual Studio
2- فهم أساسيات تطبيقات ويندوز windows applications

استكشاف بيئة التطوير المتكاملة Exploring the Integrated Development Environment

اذا كنت جاهز للعمل في ميكروسوفت فيجوال بيسك 2008, هذا الفصل سيمنحك المهارات التي تحتاجها لتنهض وتعمل مع بيئة التطوير المتكاملة للفيجوال استوديو2008 Integrated (Development Environment(IDE حيث المكان الذي ستكتب فيه برامج فيجوال بيسك,وفي هذا الفصل ستتعلم كيف تشغل فيجوال استوديو2008 وكيف تستخدم بيئة التطوير المتكاملة IDE لفتح وتشغيل برامج بسيطة, وستتعلم اساسيات قوائم واوامر بالاضافة الى اجراءات البرمجة في الفيجوال استوديو...وسوف تعمل برنامج اولي بسيط كل شي بوقته حلو...
بالاضافة الى لغة البرمجة التي ستتعلمها في هذا الكتاب, فان بيئة تطوير الفيجوال بيسك التي ستستخدمها لكتابة البرامج تدعى بيئة التطوير المتكاملة لميكروسوفت فيجوال استوديو Microsoft Visual Studio Integrated Development Environment أو تختصر الى IDE فالفيجوال استوديو ورشة عمل فعالة وقابلة للتخصيص وتحتوي على جميع الادوات التي تحتاجها لبناء برامج فعالة من اجل الويندوز والانترنت بشكل خاص.معظم المواصفات في بيئة تطوير الفيجوال استوديو تطبق تماما على الفيجوال بيسك, والسس بلس بلس C++ والسي شارب#C , استخدم التعليمات التالية لتشغيل الفيجوال استوديو 2008 :
1- اضغط Start ابدأ من شريط المهام لويندوز Windows taskbar /جميع البرامج All Programs/مجلد مكروسوفت فيجوال استوديو2008 Visual Studio) folder (2008 / ومن القائمة اختار الايقونة التي يظهر عليها شعار ميكروسوفت فيجوال استوديو2008 (Microsoft Visual Studio 2008)
ليظهر لك الاختيار السابق يجب ان تكون قد نصبت احد اصدارات الفيجوال استوديو2008 والا فلن تستطيع المتابعة,
2- اضغط على هذه الايقونة التي عليها شعار ميكروسوفت فيجوال استوديو2008 فاذا كانت المرة الاولى التي تشغل فيها فيجوال استوديو,يمكن ان تاخذ وقت لاعداد بيئة التطوير, فاذا ما طلب منك ادخال اعدادات التي ستستخدمها اختار اعدادات تطوير فيجوال بيسكVisual Basic development
عندما تبدا فيجوال استوديو سترى بيئة التطوير على الشاشة مع العديد من نوافذها,ادواتها,ونوافذ المكونات(هذه النوافذ في بعض الاحيان تدعى نوافذ الادوات *tool windows*) وسترى ايضا صفحة البداية Start Page وهي تحتوي على مجموعة من الروابط, ومواضيع شبكة مطوري مكروسوفت MSDN (Microsoft development net) صفحة البداية مصدر واسع من المعلومات حول مشاريعك بالاضافة المصادر من ضمن طائفة تطوير فيجوال بيسك والتي تربطك بمساعدة ميكروسوفت من المستندات المحلية التي تدعم بها النسخة التي نصبتها أو يمكن تحديث المعلومات مباشرة من خلال الاتصال بالانترنت كماهو مبين في الشكل.

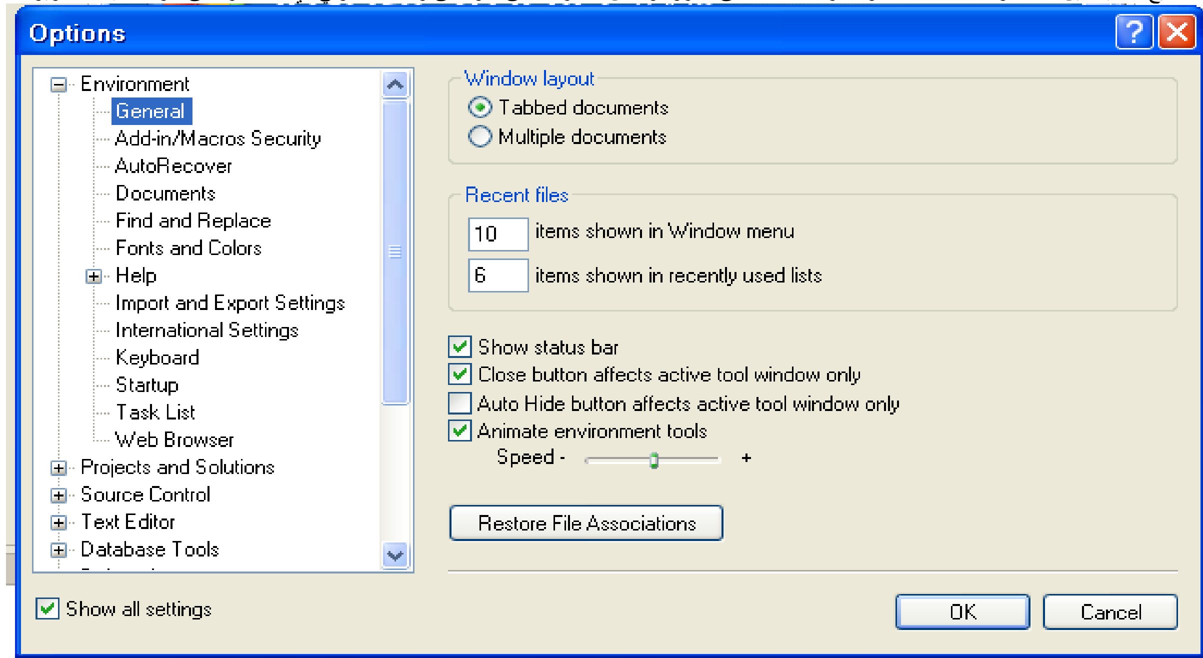


اذا وكما ذكرفي الاعلى الفيجوال بيسك هو واحد فقط من اللغات التي تستطيع استخدامها لبرمجة تطبيقاتك واللغة هي سمة واحدة فقط لتطبيق ويندوز فالواجهة المرئية للتطبيق غير مرتبطة بلغة معينة ونفس الأدوات التي ستستخدمها لتطوير واجهة تطبيقك سيتم استخدامها أيضا من قبل جميع المبرمجين بغض النظر عن اللغة التي يستخدمونها لكتابة كود التطبيق . لتبسيط معالجة تطوير التطبيقات تزودك فيجوال أستوديو ببيئة مشتركة لجميع اللغات والتي تعرف ببيئة التطوير المتكاملة integrated development environment (IDE) الهدف من (IDE) هو تمكين المطور من الاستفادة قدر الإمكان من الأدوات المرئية قبل كتابة الكود. تزود IDE بأدوات لتصميم وتنفيذ وتصحيح أخطاء تطبيقاتك ، سوف أقوم بشرح المكونات المتنوعة لبيئة التطوير كلما اقتضت الحاجة في منهج هذا الكتاب ، سوف تلقي نظرة على مكونات بيئة التطوير في هذا المقطع والتي تحتاجها لبناء تطبيقات ويندوز وسوف تتعلم كيف تسمح لك أدواتها بتصميم واجهة المستخدم الخاصة بتطبيقك بشكل سريع بالإضافة إلى كيفية برمجة التطبيق /بيئة التطوير المتكامل هي سطح مكتب ثاني سوف تقضي معظم ساعات إنتاجك في هذه البيئة .

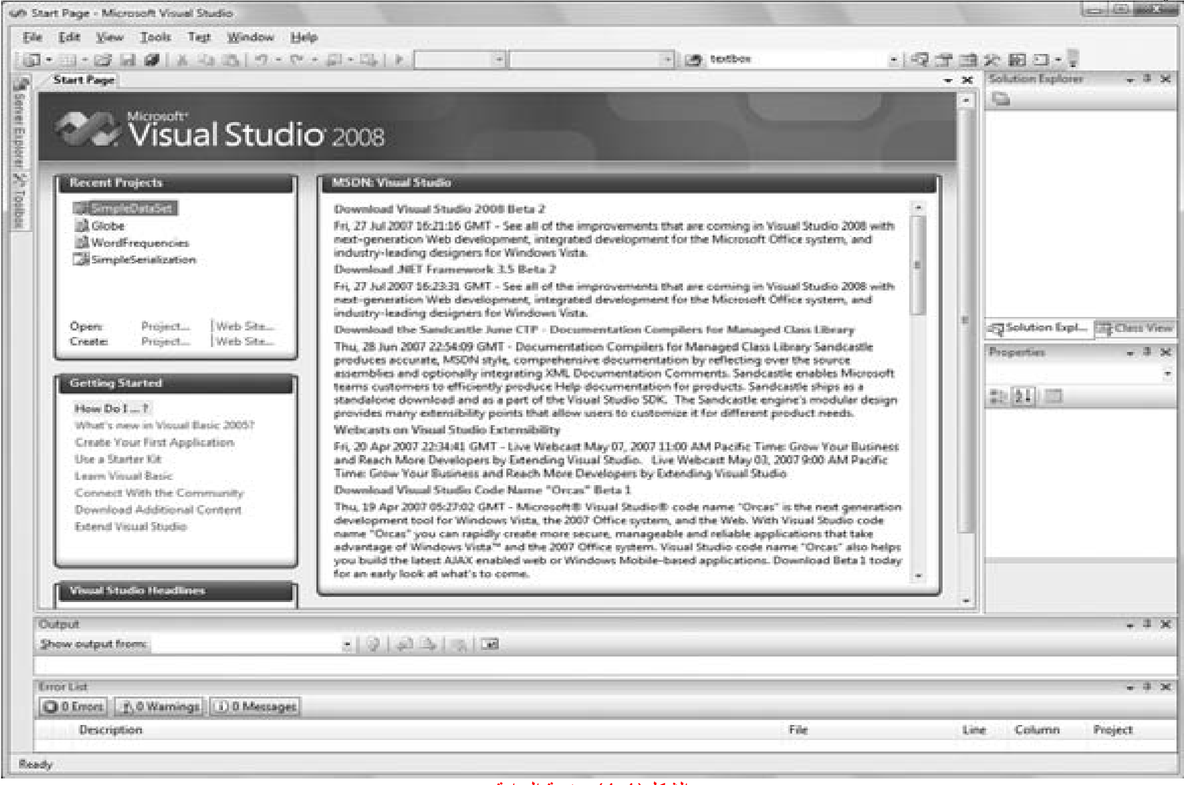
صفحة البداية The Start Page

ملاحظة اذا كنت لا ترى صفحة البداية ربما تكون بيئة التطوير لديك في عرض الوثائق المتعدد Multiple Documents بدلاً من عرض الوثائق ذات المقابض Tabbed Documents view لتبدل هذا الخيار : من قائمة ادوات Tools menu اختار الخيار الاخير Options خيارات وعلى الجهة اليسارية لصندوق حوار الخيارات Options مدد تصنيفات بيئة التطوير واضغط على General عام ومنه في اللوحة التي على اليمين لصندوق الحوار وتحت نافذة التخطيط Window Layout اضغط على الخيار وثائق مقبضية

Tabbed Documents ومن ثم اضغط OK في المرة اللاحقة التي تبدأ فيها فيجوال بيسك فان النوافذ المتعددة لبيئة التطوير قد اصبحت جميعها ذات مقابض ولديها عرى على جوانب بيئة التطوير وتستطيع التبديل بين هذه النوافذ بكل بساطة وبمجرد الضغط على العروة وما ان تغادرها حتى تعود الى وضعها الاولي أي انها تعود الى جوانب بيئة التطوير .



عندما تشغل فيجوال أستوديو لأول مرة سوف تقاد لاختيار نوع المشروع الذي تخطط لبنائه في فيجوال أستوديو لذلك فان الفيجوال أستوديو يمكن أن تخصص بشكل مثالي من اجل نوع معين من التطوير , سأفترض انك بشكل ابتدائي اخترت إعدادات تطوير فيجوال بيسك والتي تخصص نسختك من الفيجوال أستوديو لبناء تطبيقات ويندوز والويب باستخدام vb2008 تستطيع دائما تغيير هذه الإعدادات كما سيشرح في نهاية هذا المقطع ، بعد الإعداد الأولي سوف ترى نافذة مشابهة للمعروضة في الشكل الأسفل , لوحة المشروعات الحديثة ستكون فارغة ما لم تكن قد أنشأت مسبقا بعض التطبيقات , سوف يكتشف فيجوال أستوديو إعدادات من إصدار احدث لذلك إذا كنت تستخدم إصدار احدث للفيجوال أستوديو فان النافذة التمهيدية لن تكون مماثلة للظاهرة في الشكل



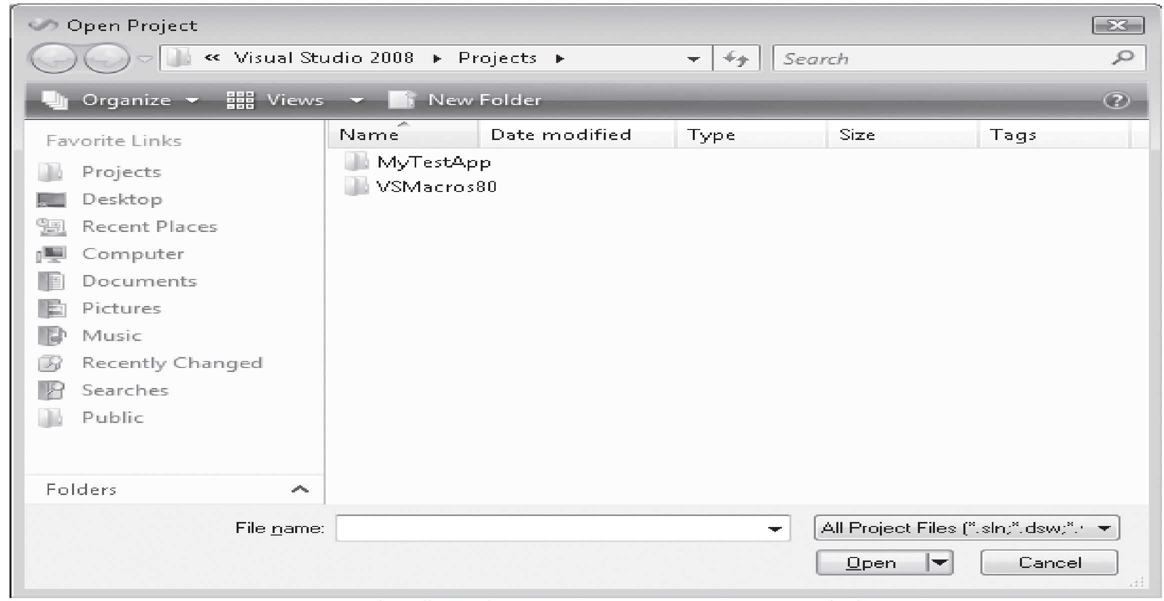
الشكل(1.1)صفحة البداية

المشروعات الحديثة Recent Projects

ترى هنا قائمة بالمشروعات التي فتحتها حديثا باستخدام فيجوال أستوديو وتستطيع اختيار أي منها لتفتحها مرة أخرى وتتابع العمل عليه

فتح مشروع فيجوال بيسك

في صفحة البداية اختارن لوحة مشاريع حديثة Recent Projects اضغط رابط link فتح مشروع Open Project , يظهر صندوق حوار فتح مشروع الظاهر في الشكل على الشاشة(تستطيع ايضا ان تعرض هذ صندوق الحوار هذا من خلال الضغط على الامر فتح مشروعOpen Project من قائمة ملف أو بواسطة الضغط على CTRL+O) وهذه النافذة مشابهة تماما لباقي نافذ الفتح في باقي البرامج تستطيع الاستعراض والبحث عن المشروع اوالملف الذي تريد من خلالها.



الشكل(1،0) صندوق حوار فتح مشروع في بيئة التطوير للفيجوال استوديو

ملاحظة : لتعرف أي من اصدارات الفيجوال استوديو تستخدم اذهب الى قائمة المساعدة (help) واضغط على الامر حول(about) حيث ان بعض المشاريع لا يمكن فتحها باصدار اقدم من الفيجوال استوديو.لذا فان المشاريع المعمولة في فيجوال استوديو2008 لا يمكن فتحها باستخدام الفيجوال استوديو2005 أو اصدار اقدم .بينما يمكن فتح التطبيقات المعمولة باصدار اقدم بواسطة اصدار أي احدث أي يمكن للفيجوال استوديو 2008 ان يفتح أي مشروع معمول باصدار اقدم سواء كان بـ2005 او2003 ...

ملاحظة : المشاريع والحلول

في الفيجوال استوديوالبرامج التي هي تحت التطوير(ما تزال في مرحلة الانشاء)بشكل نموذجي تدعى الحلول أو المشاريع projects or solutions لانها تحوي على العديد من المكونات المستقلة,وليس فقط ملف واحد,فبرامج الفيجوال بيسك2008 تتضمن ملف المشروع (.VBPROJ) project file وملف الحل(.SLN) solution file واذا ما تفحصت هذه الملفات بواسطة اداة استعراض الملفات مثل مستكشف ويندوز Windows Explorer ,ستلاحظ ان ايقونات ملف الحل عليها الرقم 9 في اعلى الايقونة كما يظهر في الشكل المرافق: يدل على رقم الاصدار النسخة التي تم برمجة هذا الحل بها.لاحظ الاحقة بعد اسم الحل في الشكل المرافق فاسم الحلcala واما الاحقة التي بعد النقطة (.sln) وكذلك بالنسبة لملف المشروع فاسم المشروع (cala) بينما الاحقة بعد النقطة هي(.vbproj) وهي اختصار لـ(visual basic project) أي مشروع فيجوال بيسك.



يشار الى الفجول بيسك 2008بشكل داخلي بـ (VB 9).يحتوي ملف المشروع على معلومات project file مخصصة لمهمة برمجية مفردة,بينما ملف الحل يحتوي على معلومات حول واحد أو اكثر من المشاريعprojects, وملفات الحل مفيدة لادارة وترتيب مشاريع متعددة مرتبطة ,وهي مشابهة لملفات تجميع المشروع project group files التي لها اللاحقة (.vbg) التي كانت في الفيجوال بيسك 6 ,الامثلة في هذا الكتاب بشكل نموذجي تحتوي على مشروع واحد single project لكل حلsolution , لذلك فان فتح ملف المشروع project file (.vbproj) أو فتح ملف الحل solution file (.sln) لهما نفس الاثر(ليس هناك أي فرق) احذر هذا فقط عندما يحتوي الحل على مشروع واحد فقط ولكن للحل المتعدد المشاريع a multi-project solution عليك فتح ملف الحل, إذا تقدم الفيجوال بيسك2008 تنسيق جديد لملف حلولها ومشاريعها .,

تزودك الفيجوال استوديو بصندوق اختيار خاص يسمى (اظهر دائما الحل Always Show Solution) وذلك من اجل التحكم بالعديد من الخيارات المتعلقة بالحلول solutions من ضمن بيئة التطوير IDE , لايجاد صندوق الاختيار هذا اختار من قائمة أدوات Tools الخيار الاخير خيارات Options ليظهر لك صندوق الحوار في الشكل ومن العروة General(عام)مدد هذه العروة تحتها تجد مشاريع وحلول Projects and Solutions / من اللوحة على اليمين شاهد هذا الخيار فاذا ما تم اختياره أو تفعيله فان مجلد فرعي سيتم انشاءه مع كل حل جديد solution ، يتم وضع المشروع وملفاته في مجلد منفصل تحت الحل,واذا ما اخترت ايضا اظهار الحل دائما فان العديد من الخيارات المرتبطة بالحلول solutions ستظهر في بيئة التطوير IDE,مثل الاوامر في قائمة ملف File ومدخلة الحل في مستكشف الحل Solution Explorer فاذا اعجبتك فكرة تحميل مجلد منفصل للحلول وتريد رؤية الاوامر والاعدادات المتعلقة بالحلول اختار صندوق الاختيار هذا,

Recent Projects
Get News from Microsoft
Options
Environment
General
Fonts and Colors
Keyboard
Projects and Solutions
Text Editor
Windows Forms Designer
Device Tools
Visual Studio projects location:
C:\Documents and Settings\mhrm\My Documents\Visual Studio 2008\Pr
Visual Studio user project templates location:
C:\Documents and Settings\mhrm\My Documents\Visual Studio 2008\Te
Visual Studio user item templates location:
C:\Documents and Settings\mhrm\My Documents\Visual Studio 2008\Te
Always show Error List if build finishes with errors
Track Active Item in Solution Explorer
Show advanced build configurations
Always show solution
Save new projects when created
Warn user when the project location is not trusted
Show Output window when build starts
Prompt for symbolic renaming when renaming files
Show all settings
OK
Cancel

شبكة مطوري فيجوال أستوديو MSDN: Visual Studio

هذا المقطع هو نافدة مستعرض والتي تعرض MSDN (the Microsoft Developer Network) والذي هو صفحة مصدر محدد لجميع تقنيات ومنتجات ميكروسوفت عندما يتصل الكمبيوتر بالانترنت في هذا المقطع سوف ترى أخبار جديدة حول الفيجوال أستوديو , اللغات المدعومة , مقالات, وأجزاء معلومات أخرى مهمة.

ابدأ Getting Started

هذا المقطع يحوي على وصلات لمهمات برمجية أساسية في منتج المستندات (التعليمات)

عناوين فيجوال أستوديو Visual Studio Headlines

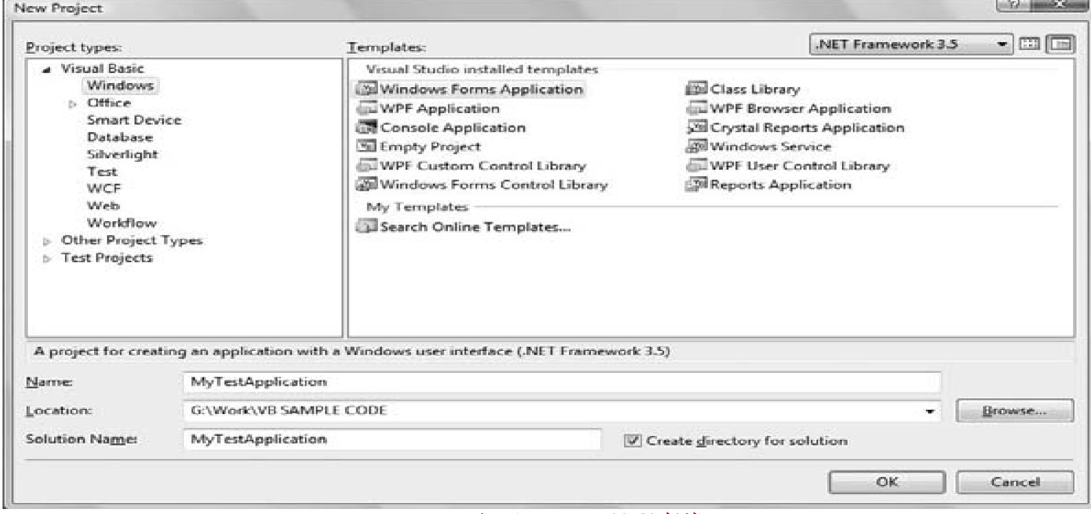
هذا المقطع يحوي وصلات لإعلانات announcements وأخبار مهمة لمطوري فيجوال بيسك
لتجعل معالج الإعداد يبدأ في صندوق Import And Export Settings واختار الأمر Tools menu معظم المطورين يتجاوزون صفحة البداية ولفعل هذا , افتح قائمة أدوات الذي ستحفظ فيه المعلومات الجديدة بحيث يستطيع الفيجوال أستوديو قراءتها في كل مرة يبدأ فيها, اترك الموقع الافتراضي كما locationالحوار الأول للمعالج والذي يسألك عن المكان مرة أخرى لرؤية النافذة الأخيرة للمعالج والتي يسال فيها لاختيار مجموعة افتراضية من الإعدادات , هذه المجموعات تعتمد على الخيارات التي تكون قد نصبتها على nextهو اضغط نظامك , أنا نصبت فيجوال استوديو2008 مع فيجوال بيسك فقط على نظامي ,وبالتالي عُرضت لي الخيارات التالية:

General Development Settings, Visual Basic Development Settings, and Web Development Settings

من الإعدادات الافتراضية لنسختي من الفيجوال أستوديو ولأهداف الكتاب اخترت إعدادات تطوير فيجوال بيسك Visual Basic Development Settings لذلك يستطيع الفيجوال أستوديو أن يكون البيئة الأمثل لتطوير فيجوال بيسك نموذجي , اضغط Finish لرؤية ملخص العملية ومن ثم أغلق المعالج.

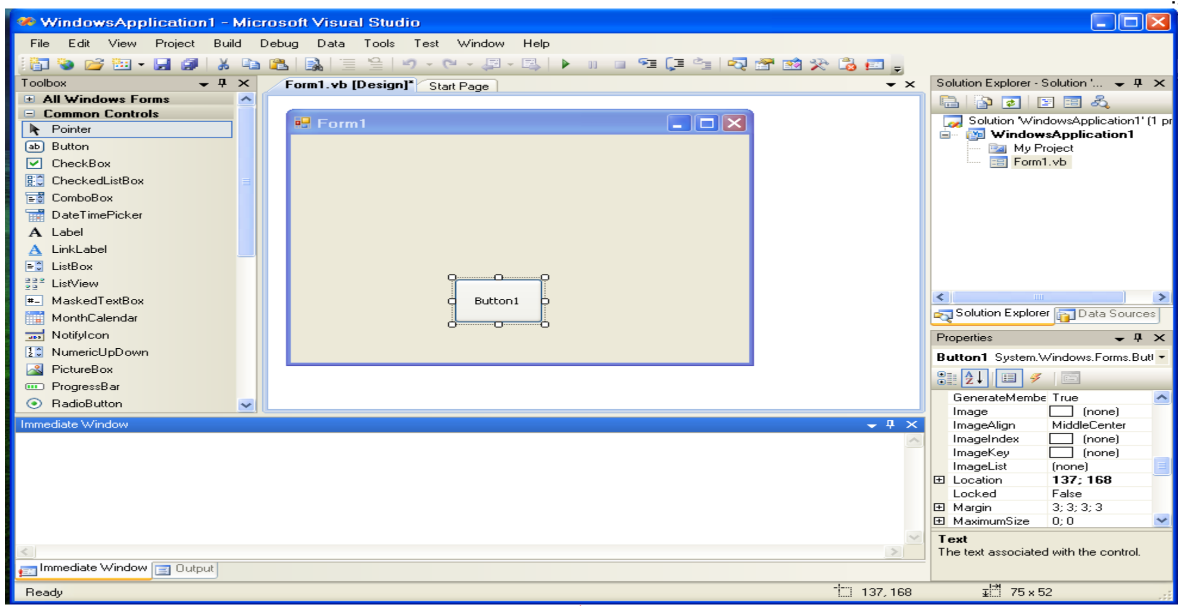
بداية مشروع جديد starting anew project

عند هذه النقطة تستطيع إنشاء مشروع جديد وتبدأ العمل في الفيجوال أستوديو , من اجل الشرح الأفضل للبنود المتنوعة لبيئة التطوير سوف نبني نموذج بسيط , وهذا النموذج هو نافذة تطبيقك (انه ما سيراه المستخدم على سطح المكتب عندما يشغل تطبيقك) افتح قائمة ملفFile menu واختار مشروع جديد New Project أو اضغط Create Project/Solution في صفحة البداية (in the Start Page) في صندوق حوار مشروع جديد الذي يظهر لك سترى قائمة بأنواع المشاريع التي تستطيع إنشاءها في الفيجوال أستوديو , الأكثر أهمية من بينها هو تطبيقات نماذج ويندوز Windows Forms Applications والتي هي تطبيق ويندوز نموذجي مع واحد أو أكثر من النماذج (forms (windows , تطبيقات الـ Console التي هي تطبيقات بسيطة والتي تتفاعل مع المستخدم من خلال نافذة نصية text window , مكتبات أدوات نماذج ويندوز Windows Forms Control Libraries التي هي مجموعات من الأدوات المخصصة ,ومكتبات الفئات والتي هي مجموعات من الفئات ، هذه هي أنواع المشاريع التي سوف نغطيها بعمق في هذا الكتاب.



الشكل(1.2)صندوق حوار مشروع جديد

إذا كنت قد نصبت فيجوال بيسك بطبعته السريعة Visual Basic 2008 Express Edition سترى عدد اقل من المشاريع في صندوق حوار مشروع جديد ولكن المشاريع التي نوقشت في هذا الكتاب ستكون محتواة لاحظ صندوق الاختيار Create Directory For Solution في الشكل السابق (صندوق حوار مشروع جديد) بشكل افتراضي يقوم الفيجوال أستوديو بإنشاء مجلد جديد للمشروع تحت المجلد الذي حددته في الصندوق Location اذا كنت تريد وضع اختصار للتطبيق بنفس المجلد لاختبار ميزات اللغة أو انجاز بعض المهمات العادية، ربما تكون لا ترغب بحفظ المشروع في هذه الطريقة عليك فقط عدم اختيار صندوق الاختيار (ازل علامة الصح من صندوق الاختيار التي تظهر في الشكل السابق) لتخطي إنشاء مجلد جديد للمشروع. تستطيع دائما حفظ المشروع في أي وقت باختيار الأمرSave All من قائمة ملف File Menu سوف تسأل عند تلك الحالة عن ملف المشروع وسيحفظ الفيجوال أستوديو المشروع تحت الملف الذي حددته انت, اذا قررت طرح الشروع جانبا discard تستطيع عندها إنشاء مشروع جديد أو إغلاق Visual Studio ستسألك الفيجوال أستوديو عن المشروع المفتوح الذي لم يتم حفظه بعد وتستطيع اختيار عدم حفظه ,
من اجل مشروعنا اختار القالب تطبيق نماذج ويندوزWindows Forms Application template سيقترح الفيجوال أستوديو لاسم WindowsApplication1 كاسم للمشروع , ضع علامة صح في صندوق الاختيار Create Directory For Solution ومن ثم اضغط ok من اجل مشروع جديد معروض في الشكل (بيئة تطوير الفيجوال أستوديو 2008 لمشروع جديد) يمكن أن تكون النافذة الرئيسية لديك مختلفة نوعا ما ولكن لا تقلق سوف استعرض جميع المكونات التي تحتاج الوصول إليها في عمليات التصميم وكتابة الاكواد واختبار تطبيقات ويندوز.



الشكل(1.3)بيئة تطوير الفيجوال أستوديو 2008 لمشروع جديد

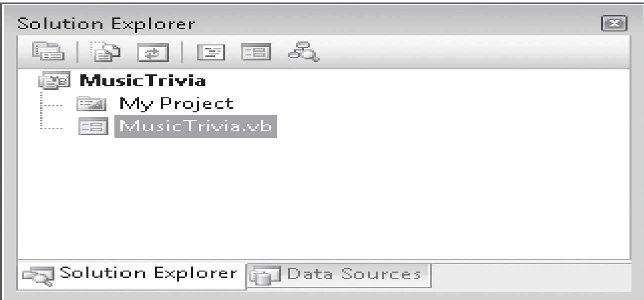
يحتوي المشروع الجديد على نموذج موجود حاليا, النموذجForm1 هو عنصر في Solution Explorer مستكشف الحل ، النافذة الرئيسية لبيئة التطوير المتكاملة IDE هي مصمم النماذج Form Designer والسطح الرمادي عليه هو نافذة تطبيقك الجديد في وضع التصميم design mode باستخدام مصمم النماذج ستكون قادر على تصميم واجهة مرئية لتطبيقك (وضع المكونات المتنوعة لواجهة الويندوز على الفورم وإعداد خواص هذه المكونات) ومن ثم برمجة التطبيق , بيئة التطوير الافتراضية نوعا ما مزدحمة لذلك دعنا نخفي عدد من أشرطة الأدوات والتي لن نستخدمها في مشاريع عدد من الفصول الأولى، تستطيع دائما إظهار أي من أشرطة الأدوات في أي وقت, افتح قائمة عرض View menu واختار شريط الأدوات

Toolbars) سوف ترى قائمة فرعية submenu مع 28 أمر التي هي (عقد اختيار أو عدم اختيار (toggles) كل أمر يتطابق مع شريط أدوات toolbars وتستطيع تحويل الشريط الموافق إلى عرض أو عدم عرض (من خلال صندوق الاختيار الذي يظهر أمامه) بواسطة الضغط مرة واحدة في القائمة الفرعية لشريط الأدوات, حاليا اختر عدم عرض جميع الأشرطة ماعدا أشرطة الأدوات toolbars (Layout and Standard) شريطا الأدوات هذان يظهران بشكل افتراضي وليس عليك إخفاؤهما اذا كنت قد أخفيتهما هذا هو المكان الذي تستطيع إظهارهما مرة أخرى.البند الأخير في القائمة الفرعية لأشرطة الأدوات Toolbars submenu هو الأمر تخصيص Customize والذي يقودك إلى صندوق حوار والذي فيه تستطيع تحديد أي من أشرطة الأدوات وأي من الأوامر التي تريد أن تراها , بعد ان تكون قد أتمت عينة العمل , اختار هذه القائمة لتخصيص بيئة التطوير للطريقة التي تريد أن تعمل بها مع الفيجوال أستوديو , تستطيع إخفاء أي من مكونات بيئة التطوير ماعدا القائمة الرئيسية, بعد كل هذا عليك أن تكون قادر على التراجع عن التغيرات التي قمت بها.

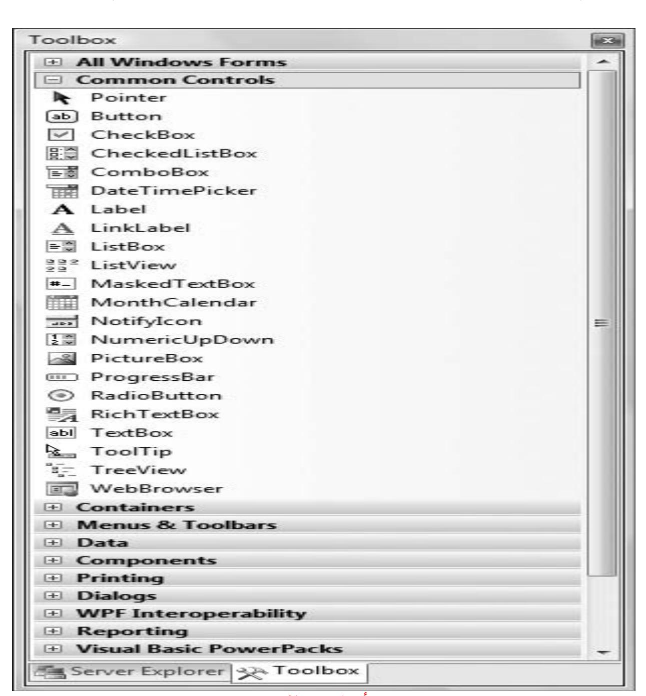
استخدام مصمم نماذج ويندوز Using the Windows Form Designer

عرض المصمم Display the Designer

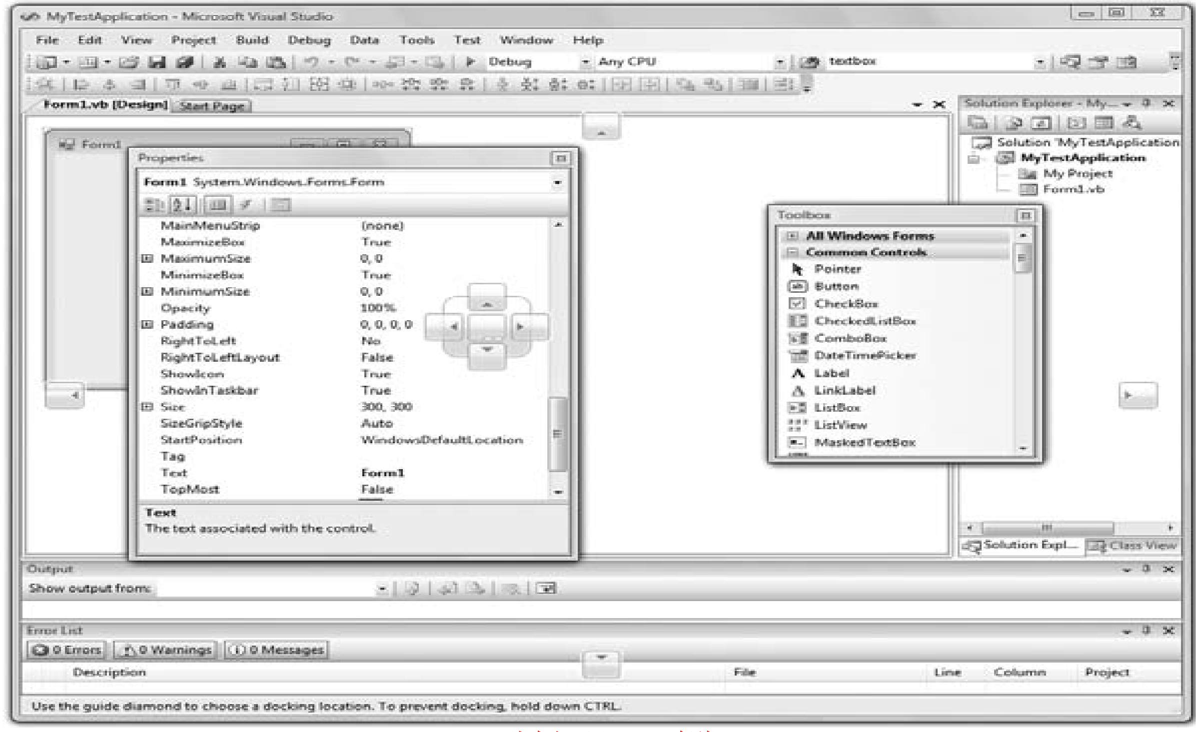
اذا كانت الفورم *form* (النموذج) أو ما تسمى واجهة المستخدم user interface (وتسمى ايضا واجهة ويندوز)غير مرئية تستطيع عرضها بشكل خاص عندما تعمل على فتح مشروع جديد كما مر معنا سابقا فيمكن الا تنتقل بيئة التطوير مباشرة الى المشروع الذي فتحته أو فورم هذا المشروع لذلك لعرض هذه النافذة اعمل التالي:
عليك ايجاد نافذة مستكشف الحلSolution Explorer وهي بالعادة تكون على الجهة اليسرى لبيئة التطوير,فاذا كانت غير ظاهرة هي ايضا من القائمة عرض(view) اختار الامر مستكشف الحل Solution Explorer وبالتالي سيظهر في المكان المخصص له على يسار بيئة التطوير فعندما يتم تحميل المشروع تظهر له ايقونة باسم المشروع الذي فتحته تحت نافذة مستكشف الحل اضغط مرتين على هذه الايقونة وسيظهر لك المشروع الذي قمت بفتحه كما في الشكل التالي لاحظ في الشكل اسم المشروع الذي قمت انا بفتحه وهو الان في مستكشف الحل واسمه MusicTrivia.vb :



لتصميم الفورم عليك وضع الأدوات عليها التي تريد عرضها للمستخدم وقت التشغيل والأدوات هي مكونات واجهة ويندوز مثل (الأزرار, صناديق النصوص, أزرار التبديل , القوائم)المعروضة في الشكل (صندوق أدوات نماذج ويندوز) صندوق الأدوات هذا يحتوي على أيقونة لكل أداة (control)تستطيع استخدامها على نموذجك (your form)، الأدوات تكون منظمة في مجموعات تبعا لوظيفة كل أداة , في هذا الفصل سوف ننشئ تطبيقات ويندوز بسيطة وسنستخدم الأدوات في لوحة الأدوات المشتركة(Common Controls tab)عندما تعمل مع تطبيقات الويب سوف ترى مجموعة مختلفة من الأيقونات في صندوق الأدوات. لوضع أداة على الفورم تستطيع بالضغط المزدوج على إيقونتها , ونسخة جديدة من الأداة مع الحجم الافتراضي سيتم وضعها على الفورم ومن ثم تستطيع تغير موضعها وحجمها بواسطة الفارة, أو تستطيع اختيار الأداة من صندوق الأدوات بواسطة الماوس ومن ثم تضغط وتسحب الماوس فوق الفورم وتحرك الماوس إلى المكان الذي ستضع فيه الأداة ومن ثم تترك الأداة وبالتالي نسخة جديدة من الأداة ستوضع على الفورم, ابدأ بوضع الأدوات على الفورم بالطريقة التي تراها مناسبة ., خواص الأداة سيتم عرضها في نافذة الخصائصProperties window (لاحظ الشكل (نوافذ بيئة التطوير)) هذه النافذة في الحافة اليمنى لبيئة التطويرIDE أسفل مستكشف الحلول(Solution Explorer) ويعرض خواص الأداة المختارة على الفورم , إذا كانت نافذة الخصائص غير مرئية , افتح القائمةView واختار نافذة الخصائص Properties Window أو اضغط F4 , إذا كان لا يوجد أداة مختارة فإنه يتم عرضه الخصائص لبند مختار في مستكشف الحل,تسمى نافذة الخصائص أيضا بمستعرض الخصائص Properties Browser التي تحدد مظهر الأداة وفي بعض الحالات وظيفتها, والخصائص منظمة تبعا لدورهم their role فالأدوات التي تتحكم في المظهر للأداة جدولت بالترتيب الهجائي تحت العنوان Appearance أما الأدوات التي تتحكم بسلوك الأداة جدولت أيضا بالترتيب الهجائي تحت العنوان header سلوكBehavior وهكذا, تستطيع ضغط الزرAZ button في شريط عنوان النافذة لعرض جميع الأدوات بالترتيب الأبجديalphabetical order , بدل الترتيب إلى الترتيب الأبجدي للخصائص.



صندوق أدوات نماذج ويندوز



الشكل(1,4)نوافذ بيئة التطوير

ملاحظة

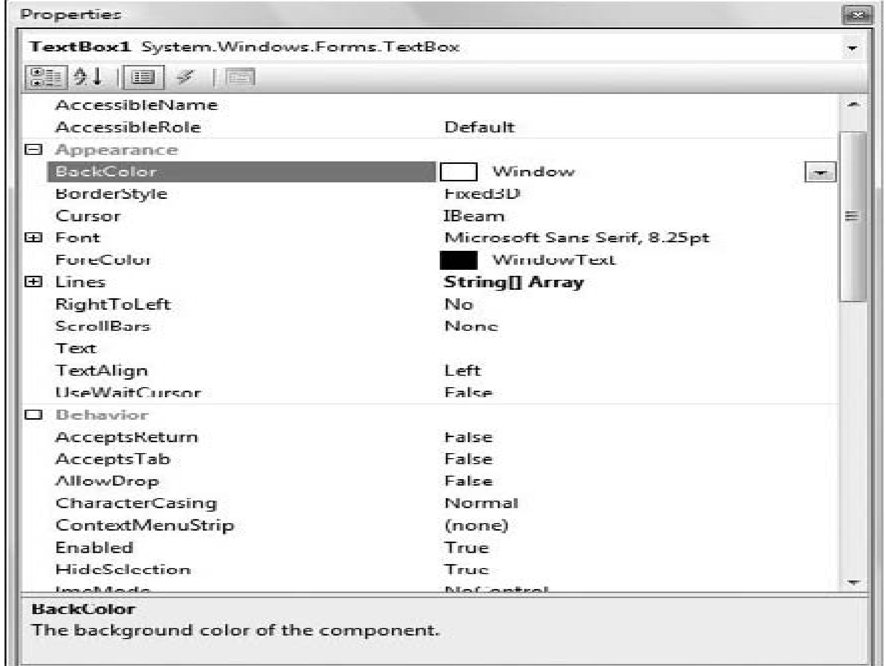
حالما تضع الأداة على الفورم فان صندوق الأدوات يتقلص إلى الحافة اليسر للمصمم retracts to the left edge of the Designer تستطيع تثبيته على الشاشة وذلك بالنقر على الإيقونة التي لها شكل القلم في شريط أدوات صندوق الأدوات (وهي الأيقونة المجاورة لإيقونة الإغلاق في الزاوية العلوية اليمنية لنافذة صندوق الأدوات Toolbox window)وهي تظهر فقط عندما يتم ترصف صندوق الأدوات(is docked) وليس عندما يكون عائم(floating) تستطيع بسهولة أن تعيد ترتيب النوافذ المختلفة لبيئة التطوير بالطريق التي تحبها (fixed, docked, floating) وما إلى ذلك.كما يظهر في الشكل (نوافذ بيئة التطوير)

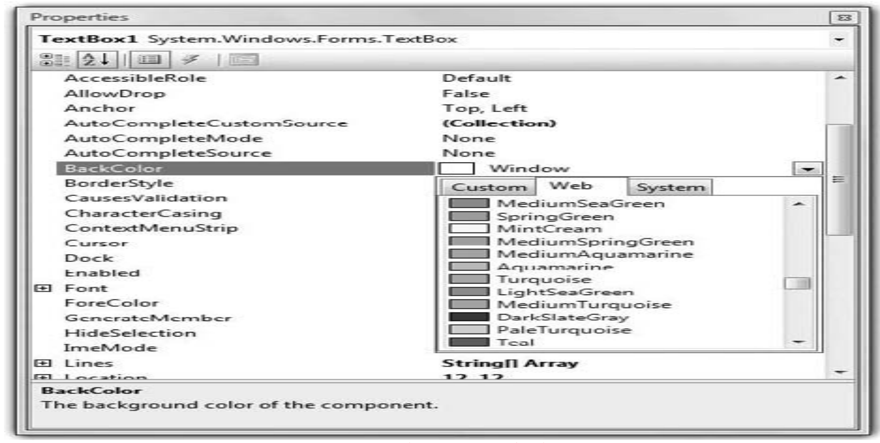
فكر بالخصائص:Thinking About Properties

في الفيجوال بيسك كل عنصر على واجهة المستخدم each user interface element في البرنامج (بما فيها النموذج نفسه اوالواجهة نفسها) لديه مجموعة من الخصائص المحددة, تستطيع ان تعد هذه الخصائص وقت التصميم باستخدام نافذة الخصائص Properties window , أو من خلال الكود نفسه لتنجز عمل هام بينما يتم تشغيل البرنامج(مثل أدوات الواجهة التي تقبل مدخلات من المستخدم غالبا ما تستخدم الخصائص لنقل المعلومات الى البرنامج).يمكن ان تجد في البداية ان الخصائص مبدا صعب الفهم,لذلك تصور انها شيء ما كاي شيء في الحياة يمكن ان يساعدك,اليك مثلا تمثيل الدراجة العادية: فالدراجة هي كائن تستخدمه لتركبه من مكان الى اخر ولان الدراجة هي كائن مادي لذا فان لديها العديد من الميزات الملازمة inherent characteristics فلديها اسم الماركة brand name ولون وسلاسل وفرامل وعجلات وكذلك تم بناءها بتصميم معين(يمكن ان تكون للرحلات,أو لصعود الجبال اويمكن ان تكون قد تم بناءها من اجل الاثنين معا) في مصطلح terminology الفيجوال بيسك هذه الميزات(التشخيصات characteristics)هي خصائص properties كائن الدراجة, معظم خصائص الدراجة تم تحديدها عندما تم بناءها,ولكن الخصائص الاخرى مثل(الاطارات, السرعة, وخيارات اخرى مثل العاكسات والمريا) هي خصائص تتغير بينما يتم استعمال الدراجة,.ويمكن حتى ان يكون للدراجة خصائص غير ملموسة intangible معنوية (والتي لا تكون مرئية) مثل تاريخ التصنيع, المالك الحالي, بيانات البيع أو الايجار وما الى ذلك ,فعندما تعمل مع فيجوال بيسك, فانك ستستخدم خاصيات الكائنات من كلا النوعين سواء كانت المعنوية أو المادية(المرئية والغير مرئية).

حدد الخاصية نصText لأداة صندوق نصTextbox control (وذلك بعد أن تضع أداة صندوق النص على الفورم وتجعلها محدد Located) أسند النص My Textbox وذلك باد خال هذا النص في الصندوق المجاور لاسم الخاصية, والخاصيةText للأداة هي النص الذي يظهر في الأداة (عنوان الأداة the control's caption)ومعظم الأدوات لديها الخاصية نص(Text property)

الموضع الذي بجانبها هو الخاصية BackColor اختارها بالفأرة, يظهر زر مع سهم صغير بجانب الإعداد الحالي للخاصية,انقر هذا الزر سوف ترى صندوق حوار مع ثلاث لوحات مرصوفةthree tabs وهي (مخصص Custom وويبWeb ونظامSystem) كما يظهر في الشكل(نافذة الخصائص) في صندوق الحوار هذا تستطيع أن تختار اللون الذي سيملأ خلفية الأداة, ضع لون خلفية الأداة إلى أصفرyellow ولاحظ أن مظهر الأداة تغير على الفورم.





الشكل(1،5)نافذة الخصائص

واحد من الإعدادات التي تريد تغييرها هي الخطfont لأدوات متنوعة بينما تكون أداة النص ماتزال مختارة على الفورم أوجد الخاصية خط في نافذة الخصائص تستطيع الضغط على(إشارة +) في مقدمة اسم الخاصية ووضع الخواص المفردة للخط , أو تستطيع الضغط على زر التبديلellipsis button لاستدعاء صندوق حوار النص ,هنا تستطيع إسناد خط الأداة ومميزاته ومن ثم تضغط ok لإغلاق صندوق الحوار , اسند خاصية الخط لأداة صندوق النصTextbox إلى (Verdana, 14 points, bold حالما تغلق صندوق حوار الخط فان الأداة على الفورم تعدل للإعدادات الجديدة

توجد فرصة ذا ما كان الخط المسند لخاصية نص الأداة control's Text property غير مناسبة لاتساع الأداة عندما تحول الخط الجديد , اختار الأداة من على الفورم بالفأرة وسترى ثمانية مقابضhandles على طول محيط الأداة perimeter ضع المؤشر فوق أي من هذه المقابض وسوق يعرض لك الشكل الذي يدلك على الاتجاه الذي تستطيع إعادة تحجيم الأداة اجعل طول الأداة مناسب لطول النص إذا كان عليك أيضا إعادة تحجيم الفورم اضغط في أي مكان على الفورم وعندما تظهر المقابض على طول محيطها اعد تحجيمها بالفأرة.

بعض الأدوات مثل Label (أداة العنوان) الزر button وصندوق الاختيارCheckBox تدعم خاصية التحجيم التلقائي والتي تحدد فيما ذا كانت الأداة التي تم تحجيمها بشكل أوتوماتيكي تلاءم عنوانها أم لا, الأداةTextBox بالإضافة إلى العديد من الأدوات الأخرى لا تدعم الخاصية AutoSize التحجيم التلقائي فإذا حاولت جعل ارتفاع الأداة مناسب accommodateلنص متعدد الأسطر, سوف تدرك انك لاستطيع تغيير ارتفاع الأداة, بشكل افتراضي أداة صندوق النص TextBox تقبل سطر مفرد للنص وعليك إعادة إسناد الخاصيةMultiLine متعدد الأسطر إلى(صح True) من اجل إعادة تحجيم الأداة عمودي vertically

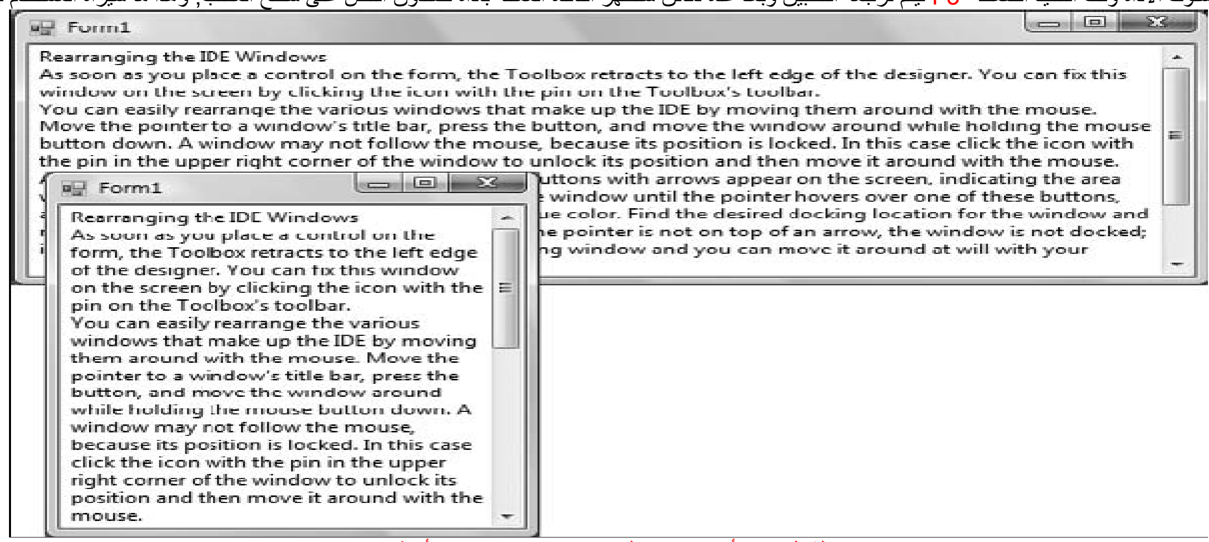
ملاحظة : الخط هو مكون تصميم وبشكل مشابه للمستندات فان النماذج (forms)يجب إن تصمم بحذر وتتبع قواعد تصميم صفحة الطباعة فعلى الأقل عليك عدم استخدام عدد من الخطوط على نماذجك كما أنه ليس عليك تحميل نموذجك أكثر مما ينبغي ولن تستخدم أيضا النمط bold style بشكل مفرطexcess , لتتجنب تعديلات خاصية الخط لعدة أدوات على الفورم سوف تقوم بإعداد خط الفورم أولا لان كل أداة يتم وضعها على النموذج ترث واو تقوم باشتقاق inherits خط الفورم فإذا ما غيرت خط الفورم فان خطوط الدودات ستتغير أيضا تبعا لذلك,بشكل عام اختار الخط الذي يناسب الواجهة التي تصممها , في كل مرة تضيف نموذج جديد الى تطبيقك عليك في البداية أن تقوم بإعداد خاصية الخط الى نفس الخط الذي استخدمته في النماذج الأخرى لذلك فان كامل التطبيق سيكون لديه مظهر مناسب consistent look الخط هو العنصر الاساسي الاكثر اهمية فيما اذا كنت تصمم مستند او فورم فالمكونات المتنوعة للفورم يمكن أن يكون لديها حجم خط مختلف وحتى نمط خط مختلف مثل غامق او مائل(bold or italics)ولكن يجب ان يكون هناك عائلة خط مسيطرة على مظهر النوذج فعائلة النمط verdana تم تصميمها لعرض الوثائق على شاشة الكمبيوتر وهو خيار عام وهناك خيار أخر هام وهو segoe ui وعائلة أنماط الخطوط الجديدة المقدمة بواسطة ويندوز فيستا خط الطباعة segoe لديه نمط تميز كتابة اليدوي handwritten ويمكنك استخدمه في التطبيقات الرسومية graphical application

مركب التصميم الثاني الأكثر اهمية هو اللون color ولكن ليس عليك أن تكون مبدع creative في مجال الألوان ما لم تكن مصمم أني اقترح ان تستخدم الألوان الافتراضية واستخدم نفس الظلال للمكونات المختلفة ,لقد اصبح تصميم الواجهات الحديثة فرع معرفي جديد new discipline لتطوير التطبيقات ويوجد أدوات لتصميم الواجهات واحدها منصة التعبير التابعة لميكروسوفتMicrosoft's Expression Studio والتي تمكن المصممين من تصميم الواجهة interface والمطورين من كتابة الكود , بدون قطع عمل أي منها بواسطة لأخر تستطيع إنزال إصدار تجريبي من Expression من www.microsoft.com/expression

حتى لان لقد قمت بمعالجة الخصائص التي تحدد المظهر للأداة , الآن سوف تغير الخصائص التي تحدد ليس فقط المظهر ولكن أيضا وظائف function الأداة حدد الخاصية Multiline والإعداد الحالي لها هو تعطيل False مدد قائمة الإعدادات الممكنة غيرها إلى True, (تستطيع أيضا تغيرها بواسطة الضغط المزدوج على اسم الخاصية) وهذا الفعل هوtoggles عقدة اختيار والتي تتبدل بالضغط إلى (صح /خطأ) ارجع إلى الفورم واختار الأداة TextBox واجعلها طويلة كما ترغب . الخاصيةMultiline تحدد فبماذا الأداة TextBox تقبل سطر أو أكثر (إذا كانت الخاصية(Multiline = False) أو أكثر إذا كانت الخاصية (if Multiline = True) ضع هذه الخاصية إلى صح True ارجع إلى خاصية النص Text واكتب فيها نص طويل , واضغط Enter ستقوم الأداة بتقسيم النص الطويل إلى عدة اسطر . إذا قمت بإعادة تحجيم الأداة فان الأسطر ستتغير ولكن كامل النص سيتناسب مع الأداة لأن خاصية التفاف النص للأداة WORDWRAP هي true اسند لها False لترى كيف سيتم تحويل النص على الأداة.

أداة صندوق النص المتعدد الأسطر لديها خاصية شريط منزلقة عمودية Scroll Bar لذلك يستطيع المستخدم إيجاد مقطع النص الذي يريده بسرعة وبسهولة , اوجد الخاصية Scrollbarsللأداة ومدد القائمة الممكنة للإعدادات بواسطة الضغط على الزر الذي عليه إشارة السهم arrow وإعدادات هذه الخاصية هي (None, Vertical, Horizontal, and Both) والتي تعني (بدون، عمودي, أفقي ,كلاهما)على الترتيب. أسند لها Vertical, اكتب نص طويل جدا في خاصية النص لها وراقب كيف تعامل الأداة النص لا تستطيع زحلقة النص على الأداة وقت لتصميم , ولكن شريط الانزلاق يعمل كما هو متوقع وقت التنفيذ(سوف يزلف النص بشكل عمودي) , تستطيع أيضا أن تجعل الأداة تملأ كامل النموذج , ابدأ بحذف جميع الأدوات الأخرى على النموذج والتي يمكن أن تكون قد وضعتها ومن ثم اختار صندوق نص متعدد الأسطر(multiline TextBox) اوجد الخاصيةDock في نافذة الخصائص Properties window واضغط مزدوج على اسم الخاصية حتى تتغير إعداداتها إلى ملئ Fill أداة النص ستقوم بملأ النموذج وتم تحجيمها على الفورم في كلا الحالتين وقت التصميم والتنفيذ.

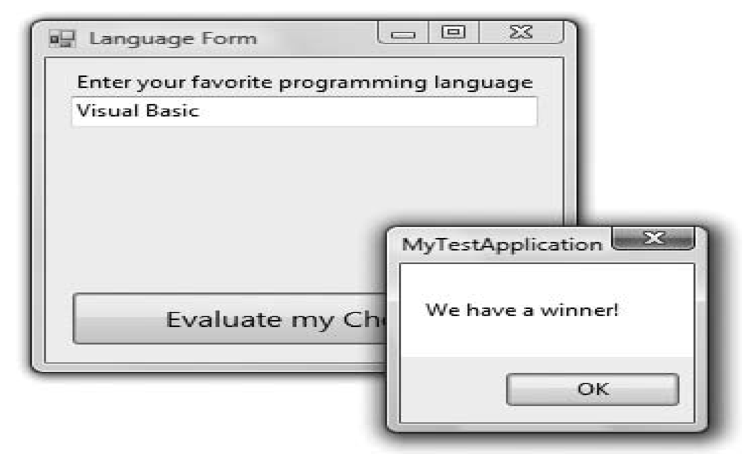
لتختبر سلوك الأداة وقت التنفيذ اضغط F5 ليتم ترجمة التطبيق وبعد عدة دقائق ستظهر النافذة المملئة بأداة صندوق النص على سطح المكتب, وهذا ما سيراه المستخدم لتطبيقك



الشكل(1،6)أداة صندوق النص تعرض نص متعدد الأسطر

إنشاء تطبيق الأول في الفيجوال بيسك Creating Your First VB Application

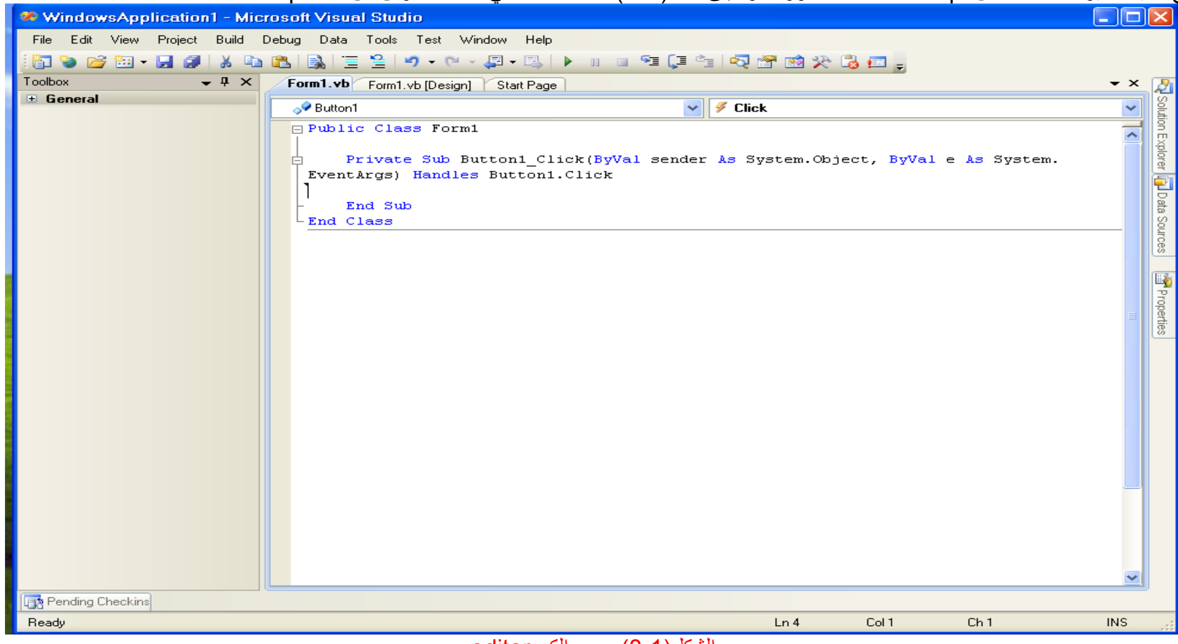
في هذا الفصل تقوم بتطوير تطبيق بسيط لتوضيح ليس فقط تصميم الواجهة ولكن أيضا كتابة الكود خلف الواجهة , سنقوم ببناء تطبيق يسمح للمستخدم بإدخال اسم لغة البرمجة المفضلة وسيقوم التطبيق من التحقق من الاختيار, بشكل موضوعي VB هي في أعلى جميع اللغات الأخرى وسوف تتلقى التقييم الأفضل بينما جميع اللغات الأخرى ستتلقى نفس الدرجة وهي good وليس VB , أبدأ مشروع جديد new project واستخدم نفس الاسم الافتراضي **Windows Application1** وضع الأداة TextBox **والأداة** Button على النموذج(form) استخدم الفأرة لوضع وإعادة تحجيم الأدوات على الفورم **كما يظهر في الشكل المرفق**



الشكل (1،7)تطبيق بسيط والذي يعالج النص المدخل من قبل المستخدم

أبدأ بإعداد خاصية الخط للفورم Font إلى Segoe UI, 9 pt (سيغو, 9 نقطة) نظم وحجم الأدوات كما يظهر في الشكل السابق ومن ثم ضع أداة عنوان على الفورم وأسند في خاصية النص Text النص التالي(**Enter your favorite programming language**)سيتم إعادة تحجيم أداة العنوان تبعا لعنوانها لان خاصية الأداة AUTOSIZE هي صح(True),كلما حركت الأدوات داخل الفورم سترى بعض الخطوط الزرقاء التي تصل حواف الأدوات عندما يتم عمل ترصيف للأدوات aligned هذه الخطوط تسمى خطوط الترصيف snap lines والتي تسمح لك بترتيب الأدوات على الفورم , الآن عليك إدخال الكود الذي يتحقق من لغة المستخدم المفضلة ,تطبيقات ويندوز مركبة من مقاطع كود صغيرة تدعى معالجات الحدث event handlers والتي تستجيب إلى فعل معين مثل نقر الزر click of a button , اختيار قائمة أمر the selection of a menu command , النقر على صندوق اختيار the click of a check box , وهكذا .- عندما ينقر المستخدم الزر,نريد أن ينفذ بعض الكود الذي يعرض رسالة ما.

لإدخال بعض الكود خلف أداة الزر اضغط مزدوج على هذه الأداة وسترى نافذة كود التطبيق , والتي تظهر في الشكل الأسفل , سوف ترى فقط تعريف الإجراء, إذا كان سطر الكود طويل جدا تستطيع تقسيمه وذلك بإدخال خط منخفض underscore في المكان الذي تريد قطع السطر عنده وبالتالي سيعتبر الفيجوال بيسك كلا السطرين سطر واحد,أو خط كود واحد, وبشكل اختياري تستطيع تبديل الميزة التفاف النص Word Wrap لمحرر الكود إلى on (تعمل) كما فعلت انا في الشكل المرفق من قائمة edit>advance>word wrap .



الشكل(1،8) محرر الكودeditor

يفتح محرر الكود الأجراء الفرعي المحدد بالعبارات التالية:

```
Private Sub Button1 Click(ByVal sender As System.Object, _
        ByVal e As System.EventArgs) Handles Button1.Click

End Sub
```

سترى ف أعلى لوحة اللوحة الرئيسية للمصمم عروتين مسماة بعد اسم الفورم وهي عروة Form1.vb [Design] وعروة the Form1.vb العروة الأولى هي عروة مصمم نماذج ويندوز (التي تستطيع بها بناء واجهة التطبيق بواسطة الأدوات المرئية) والثانية هي محرر الكود (التي بها تدخل الكود خلف الواجهة) في أعلى محرر الكود والذي رأيته في الشكل السابق هناك صندوقي تركيب الأول إلى اليسار ويحتوي على أسماء الأدوات التي على الفورم والآخر الذي على اليمين يحتوي على أسماء الأحداث لكل أداة معروفة, عندما تختار الأداة (أو أي كائن بشكل عام)في القائمة التي على اليسار فان محتوى القائمة الأخرى التي على اليمين يعدل تبعا لهذه الأداة. لبرمجة حدث معين لأداة معينة اختار اسم الأداة في القائمة اليسارية (قائمة الكائنات) واسم الحدث من القائمة التي على اليمين(قائمة الأحداث) بينما يكون الزر Button1 تم اختياره في قائمة الكائنات Objects list افتح قائمة الأحداث Events list لترى الأحداث التي يستجيب لها الزر button

الحدث Click يحدث لأن يكون الحدث الافتراضي لأداة الزر Button control لذلك عندما تضغط مزدوج على الزر الذي على الفورم سيتم أخذك إلى الإجراء الفرعي Button1_ Click وهذا الإجراء الفرعي هو معالج الحدث والذي يتم استدعاؤه بشكل اوتماتيكي في كل مرة يحدث حدث ما . الحدث الذي يهمنا في مثالنا هو الحدث نقر أو click لأداة الزر Button1 في كل مرة يتم فيها نقر أداة الزر على الفورم فان الأجراء الفرعي Button1_ Click يتم تفعيله , ليستجيب إلى حدث نقر الزر , يتوجب عليك إدخال الكود المناسب في هذا الإجراء الفرعي , يوجد أيضا أكثر من 24 حدث لأداة الزر والأداة Button control هي واحدة من أبسط الأدوات, معظم الأدوات تعرف عدد كبير من الأحداث events .

تعريف معالج الحدث لايمكن تعديله, وهذا التعريف هو الـsignature أو دليل معالج الحدث (أو المعاملات النسبية arguments التي يمررها الحدث إلى التطبيق) جميع معالجات الحدث في VB 2008 تمرر معاملان نسبيان two arguments إلى التطبيق وهما: المعامل النسبي sender argument والذي هو كائن يمثل الأداة التي أطلقت الحدث , والمعامل النسبي e argument الذي يعطي بمعلومات إضافية حول الحدث

اسم الإجراء الفرعي مركب من اسم الأداة متبوعا بخط منخفض واسم الحدث ,هذا فقط الاسم الافتراضي وتستطيع تغيره إلى أي اسم تراه مناسبا(مثل EvaluateLanguage) من أجل هذا المثال , ما يجعل الإجراء الفرعي معالج للحدث هو الكلمة المحجوزة Handles عند نهاية العبارة . الكلمة المحجوزة Handles تخبر المترجم أي حدث هذا الإجراء الفرعي يقترح معالجته, ف Button1.Click هو حدث نقر الأداة Button1 , فإذا ما كان هناك زر أخر على الفورم, الأداة Button2 عليك كتابة الكود للإجراء الفرعي الذي عليه معالجة الحدث Button2.Click,كل أداة لديها العديد من الأحداث ,وتستطيع أن تزود كل أداة بمعالج حدث ما مختلف وحدث مصاحب لها(من مجموعة أحداثها),بالطبع لن نقوم ببرمجة كل حدث ممكن لكل أداة .

إن الأدوات لديها سلوك افتراضي وتعالج الأحداث الأساسية الخاصة بها, فالأداة TextBox تعرف كيفية معالجة الضرب keystrokes على المفاتيح, الأداة CheckBox (التي هي مربع بداخله علامة اختيار(صح)) تغير الحالة وذلك بإخفاء أو إظهار علامة الصح (الاختيار)في كل مرة يتم نقرها, أداة شريط الانزلاق ScrollBar تحرك مؤشرها في كل مرة تضغط واحد من الأسهم عند كلا النهايتين , بسبب هذا السلوك الافتراضي للأدوات لا تحتاج إلى تزويدها بالكود من اجل أغلب أحداث الأدوات على النموذج(الفورم).إذا غيرت اسم الأداة بعد أن تكون قد أدخلت بعض الاكواد في معالج الحدث فان اسم الحدث الذي تم معالجته بواسطة الإجراء الفرعي سوف يتغير بشكل أوتوماتيكي , بينما اسم الإجراء الفرعي مهما يكن لن يتغير,فإذا ما غيرت اسم الأداة Button1 إلى **bttnEvaluate** فان عنوان الإجراء الفرعي سيصبح:

```
Private Sub Button1_ Click(ByVal sender As System.Object, _
                        ByVal e As System.EventArgs) Handles bttnEvaluate.Click
End Sub
```

أعد تسمية الإجراء الفرعي إلى ,يتوجب تحرير الكود لتغير اسم معالج الحدث , أنا قمت بتسمية الأدوات قبل إضافة أي كود للتطبيق ,لذلك فان معالج الحدث الخاص بهم سيتم تسميته بشكل صحيح ,بشكل اختياري استخدام أسماء تخصصها أنت لكل معالج الأسماء الافتراضية للأدوات التي وضعتها على الفورم هي أسماء عامة وسوف تغيرها إلى شيء ما له معنى, أنني عادة اسبق أسماء الأدوات بعدة أحرف والتي تدل على نوع الأداة (مثل txt, lbl, bttn) متبوعة بأسماء لها معنى, أسماء مثل txtLanguage و bttnEvaluate تجعل كودك أكثر قابلية للقراءة, انه تدريب جيد أن تقوم بتغير الأسماء الافتراضية للأدوات حالما تضيف أدوات إلى الفورم فأسماء مثل Button1, Button2, Button3 لا ترقى promote ان تكون قابلة للقراءة في كودك , باستثناء هذا التطبيق فاني استخدمت أسماء أكثر قابلية للقراءة للأدوات التي استخدمت في باقي الكتاب .

دعنا نضيف بعض الكود إلى معالج الحدث Click لأداة الزر Button1 عندما يتم ضغط هذا الزر نريد أن نختبر النص في صندوق النص, فإذا كان Visual Basic نعرض رسالة للمستخدم وإذا لم تكن كذلك نعرض رسالة مختلفة ادخل الأسطر المعروضة في القائمة التالية بين العبارتين Sub Private و End Sub (كامل القائمة معروضة هنا لإعادة كتابة العبارات الأولى والأخيرة)

```
Private Sub EvaluateLanguage(ByVal sender As System.Object, _
                        ByVal e As System.EventArgs) Handles Button1.Click
Dim language As String
language = TextBox1.Text
If language = "Visual Basic" Then
MsgBox("We have a winner!")
Else
MsgBox(language & "is not a bad language.")
End If
End Sub
```

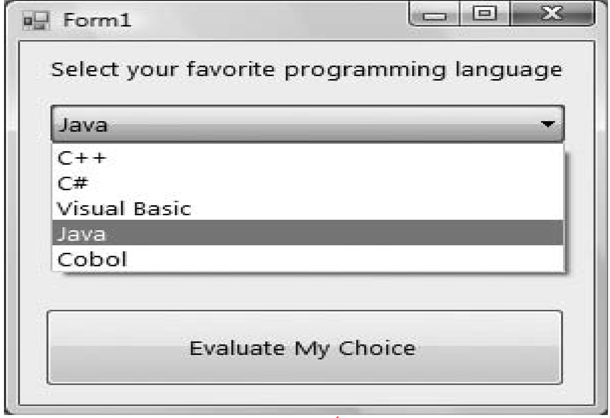
إليك ما يعمله الكود في البداية يقوم نص الأداة TextBox إلى المتغير language والمتغير variable هو مكان محجوز في الذاكرة (named location) والذي تخزن فيه القيمة. المتغيرات هي المكان الذي نخزن فيه نتائج الوسائط(معالات) لحساباتنا عندما نكتب الكود, يتم التصريح عن جميع المتغيرات بواسطة العبارة dim ولديها أيضا اسم ونوع, تستطيع أيضا أن تصرح وتسند قيمة للمتغير language بخطوة واحدة كما هو مبين في العبارة التالية: Dim language = TextBox1.Text

سينشأ المترجم متغير نصي لان العبارة تسند نص الى المتغير (سوف ندرس مسألة الإعلان عن المتغيرات فيما بعد), ومن ثم يقارن البرنامج قيمة المتغير language variable بالعبارة Visual Basicوبالاعتماد على مخرجات المقارنة يعرض واحد من الرسالتين بنافذة صغيرة مع الزر OK فقط (كما هو واضح في الشكل رقم 8) يستطيع المستخدمون عرض الرسالة ومن الضغط على الزر OK لاغلاق صندوق الرسالة, اذا كنت لا تعلم تركيب اللغة عليك ان تكون قادر على معرفة ما يعمله هذا الكود فلغة الفيجوال بيسك هي ابسط لغة مدعومة من قبل الفيجوال استوديو2008 وسوف نناقش السمات المتنوعة لهذه اللغة بالتفصيل في الفصول اللاحقة, في الوقت الحالي عليك المحاولة على فهم عمليات تطوير تطبيقات ويندوز ,كيفية بناء واجهة التطبيق وكيفية برمجة الأحداث التي تريد ان يستجيب لها تطبيقك. الكود في تطبيقنا الأول غير نشيط بشكل كافيvery robust . فإذا لم يدخل المستخدم نفس العبارة وبنفس التهجية الظاهرة في عبارة الكود فان عملية المقارنة ستفشل نستطيع تحويل النص الى حالة الأحرف الكبيرة ومن ثم نقوم بمقارنتها بالعبارةVISUAL BASIC للخلص من الإختلاف في حالة الأحرف لتحويل العبارة إلى حالة الأحرف الكبيرة استخدم الطريقة ToUpper method من فئة النص String class التعبير التالي يعيد بالنص المخزن في المتغير language variable ويحوله إلى الحالة الكبيرةuppercase للأحرف language.ToUpper

سنأخذ أيضا بعين الاعتبار حقيقة أن المستخدم يمكن أن يدخل VB أو VB 2008, في المقطع التالي سنقوم بتحسين تطبيقنا , في حالتنا يمكن أن نعرض أسماء لغات محددة (اللغات التي تهمنا) ونجبر المستخدم على اختيار واحد منها. واحدة من الطرق التي تستخدم لعرض عدد محدد من الخيارات هو استخدام اداة صندوق مركبComboBox في المقطع التالي سنعمل على تعديلrevise تطبيقنا البسيط لذلك فان المستخدم ليس عليه ادخال اسم اللغة سنجبرهم على اختيار لغتهم المفضلة من قائمة لذا ليس علينا التقق من النص المدخل بواسطة المستخدم.

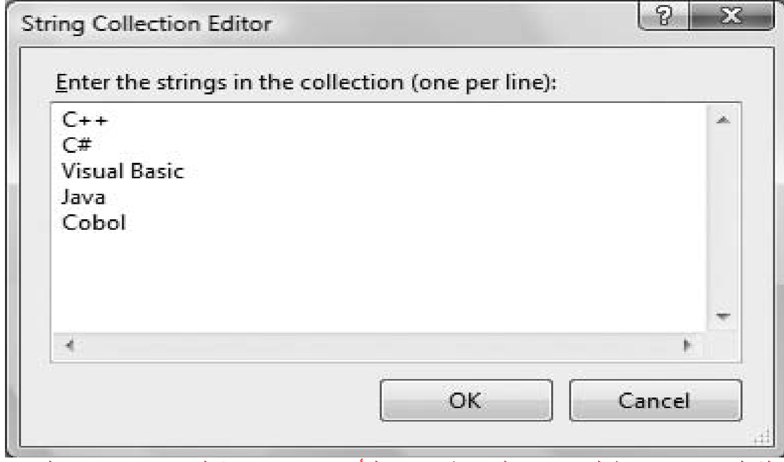
جعل التطبيق أكثر سهولة للمستخدم Making the Application More User-Friendly

ابدأ مشروع جديد (WindowsApplication2) كما تعلمت سابقا , ولا تختار صندوق الاختيارCreate Directory For Solution سنحفظ المشروع من خلا بيئة التطويرIDE حالما يتم إنشاء المشروع, افتح قائمة ملفFile menuواختارSave All لحفظ المشروع وعندما يظهر صندوق حوار حفظ المشروع انقر الزر Browse لاختيار المجلد الذي ستحفظ المشروع فيه اختار مجلد موجود أو انشيء مجلد جديد , افتح صندوق الأدوات واضغط مزدوج على أداة صندوق المركب ComboBox سيتم وضع الأداة على الفورم والآن ضع أداة الزرButton على الفورم بحيث تبدو الفورم مثل الشكل(1,9) اسند النص.**Evaluate My Choice** إلى خاصية النص text للزر



الشكل(1,9)عرض خيارات أداة الصندوق المركبComboBox

يتوجب علينا الآن مليء الأداة صندوق الاختيار بالاختيارات الممكنة, اختار أداة ComboBox من على الفورم بواسطة الضغط بالفأرة عليها و اوجد الخاصية Itemsمن نافذة الخصائص إعدادات هذه الخاصية هي مجمع Collection والتي تعني أن الخاصية Items ليس لديها قيمة وحيدة فهي مجموعة من القيم (نصوص في حالتنا) اضغط زر التبديل وسوف ترى صندوق حوار مجمع النص كما يظهر في الشكل



الشكل(1،10) اضغط الزر بجانب الخاصية items للأداة ComboBox لعرض مدير مجمع النص

اللوحة الرئيسية لصندوق حوار مدير مجمع النص هي text box صندوق نص والتي تمكنك من إدخال البنود التي تريد إظهارها في أداة الصندوق المركبComboBox وقت التنفيذ, قم بإدخال النصوص التالية واحد في كل سطر وبالترتيب الظاهر هنا :

C++
C#
Visual Basic
Java
Cobol

اضغط الزرok لإغلاق صندوق الحوار , البنود لن تظهر ف الأداة وقت التصميم , ولكن ستراها وقت التنفيذ , قبل تنفيذ البرنامج قم بإعداد بعض خاصيات الأداة صندوق النص ,قم بإيجاد الخاصية text له واكتب فيها النص التالي : (select your favorite programming language)وهذه ليست بند من القائمة السابقة إنما هي النص والذي سيظهر بشكل أولي على الأداة.

تستطيع الأن تشغيل المشروع وترى سلوك أداة الصندوق المركب, اضغط F5 وانتظر عدة ثواني سيتم ترجمة المشروع وسترى نموذجه على سطح المكتب, تستطيع اختيار بند من بنود الأداة إما بالفأرة أو بلوحة المفاتيح أو بضغط مفاتيح الأسهم للأعلى أو للأسفل لتتنقل ضمن قائمة البنود, اضغط المفتاح tab لنقل التركيز إلى أداة الزر واضغط مفتاح المسطرة spacebar (أو ببساطة انقر على أداة الزر) نحن لم نخبر الزر ما عليه فعله عند الضغط عليه , لذا دعنا نعود لإضافة بعض الكود إلى المشروع , أوقف تنفيذ المشروع بالضغط على زر الإيقاف في شريط الأدوات toolbar(المربع الأسود) او باختيار debug>stop debugging من القائمة الرئيسية . عندما تظهر الفورم في وضع التصميم اضغط مزدوج على أداة الزر وسوف تفتح نافذة الكود , عارضة معالج حدث الضغط وهو فارغ, قم بادخال العبارات التالية بين Private Sub و end sub

```
Private Sub Button1 Click(ByVal sender As System. Object, _
                        ByVal e As System.EventArgs) Handles Button1.Click
Dim language As String
language = ComboBox1.Text
If language = "Visual Basic" Then
MsgBox("We have a winner!")
Else
MsgBox(language & "is not a bad language.")
End If
End Sub
```

عندما يتم عرض الفورم للمرة الأولى سيتم عرض نص لا يتطابق مع أي من اللغات المعروضة في الأداة ComboBox نستطيع أن نختار بشكل مسبق واحد من البنود من ضمن كودنا عندما يتم تحميل الفورم فانه يتم إطلاق حدث تحميل كائن الفورم , اضغط مزدوج في مكان ما على الفورم وبالتالي فان محرر الكود سيفتح معالج حدث تحميل الفورم .

```
Private Sub Form1 Load(ByVal sender As System.Object, _
                        ByVal e As System.EventArgs) Handles MyBase.Load
End Sub
```

ادخل الكود التالي لاختيار البند Visual Basic عندما يتم تحميل الفورم.

```
Private Sub Form1 Load(ByVal sender As System. Object, _
                ByVal e As System.EventArgs) Handles MyBase.Load
        ComboBox1.SelectedIndex = 2
End Sub
```

إن SelectedIndex هو خاصية للأداة ComboBox والتي تحدد البند المختار, تستطيع إسنادها إلى قيمة عددية صحيحة من ضمن كودك لاختيار البند على الأداة وتستطيع استخدامها لاستخلاص فهرس البند المختار في القائمة , فبدلا من مقارنة النصوص نستطيع ان نقارن الخاصية SelectedIndex بقيمة تطابق الفهرس لبندVisual Basic بعبارة مثل التالية:

```
If ComboBox1.SelectedIndex = 2 Then
    MsgBox("We have a winner!")
Else
    MsgBox(ComboBox1.Text & "is not a bad language.")
End If
```

الخاصية text لأداة ComboBox ترجع بالنص الموجود على الأداة ونستخدمه لطباعة اسم اللغة المختارة, بالطبع إذا ما أدخلت أو حذفت بند من القائمة يتوجب عليك تحديث الكود طبقا لذلك, إذا نفذت التطبيق واختبرته ستدرك أن هناك مشكلة في أداة ComboBox, حيث أن المستخدم يستطيع إدخال نص جديد في الأداة والذي سيترجمinterpreted كلغة , بشكل افتراضي أداةComboBox تسمح للمستخدم بكتابة شيء ما بالإضافة لاختيار بند من القائمة, لتغير سلوك الأداة, اختارها من على الفورم وأوجد الخاصية التابعة لها DisplayStyle في نافذة الخصائص مدد قائمة الإعدادات الممكنة للأداة وغير قيمة الخاصية من DROPDOWN إلى DROPDOWNLIST نفذ البرنامج مرة أخرى واختبره, لقد أصبح تطبيقنا متين bulletproof وهو تطبيق بسيط سوف ترى تقنيات أكثر لبناء تطبيقات قوية robust في الفصل الرابع.

فهم مكونات بيئة التطوير المتكاملة Understanding the IDE Components

تحوي IDE لـVisual Studio 2008 على عدد ضخم من المكونات, وستأخذ منك وقت لاستعراض مكوناتها ومن المستحيل شرحها في فصل واحد لذا سنناقش أدوات خاصة ومن ثم وخلال المواضيع المختلفة في الفصول اللاحقة سيتم تغطيت ما أمكن من هذه المكونات, البنود الأساسية التي سنستخدمها في الفصول القليلة القادمة هي في بناء تطبيقات ويندوز بسيطة

قوائم بيئة التطوير The IDE Menu

قوائم بيئة التطوير تزودك بالأوامر التالية والتي تقود إلى قوائم فرعية, لاحظ أيضا أن معظم القوائم يمكن أن تعرض أيضا كأشرطة أدوات toolbars وليس كل الخيارات متاحة في كل الأوقات فالخيارات الغير المتاحة للوضع الحالي من بيئة التطوير هي إما غير مرئية أو غير ممكنة(معطلةdisabled) فقائمة تحريرedit menu هي مثال نموذجي فتكون قصيرة تماما

عندما تكون في وضع تصميم الفورم وطويلة نوعا ما عندما تحرر الكود, القائمة data menu لا تظهر على الإطلاق altogether عندما تفتح محررا لكود ولا تستطيع استخدام خيارات هذه القائمة عندما تكون في محرر الكود , إذا فتحت مستند XML في بيئة التطوير فان أوامر xml ستضاف إلى قائمة فيجوال أستوديو الرئيسية

قائمة ملف File Menu

تحتوي على أوامر لفتح وحفظ المشاريع أو البنود بالإضافة إلى أوامر لإضافة بند جديد أو موجود للمشروع الحالي جميع خيارات هذه القائمة معروفة للجميع وهي نفسها في معظم البرامج تقريبا بالإضافة إلى وجود الأمر save all والذي يعمل على حفظ جميع مكونات المشروع الحالي -----) استعرض هذه القائمة واستكشف ما تحتويه من أوامر.

القائمة تحرير Edit Menu

من بين الأوامر التي تحويها هذه القائمة هي أوامر متقدمة advanced والأمر IntelliSense كلا الأمرين يقودان الى قوائم فرعية ولاحظ ان هذين البندين مرئيين فقط عندما تحرر الكود وغير مرئيان أثناء تصميم الفورم

القائمة تحرير < القائمة الفرعية (خيارات متقدمة) Edit > Advanced Submenu

الخيارات الأكثر أهمية لقائمة تحرير ومنها القائمة الفرعية **Advanced** هي التالية

View White Space عرض مساحة فارغة (محارف الفراغ Space characters) الضرورية لترك فراغ لكود وجعله سهل القراءة) والتي تستبدل بواسطة النقاط periods

التفاف النص **Word Wrap** عندما يكون سطر الكود طويل ويزيد عن الطول في نافذة الكود فان الخطوط سوف تلتف بشكل آلي .

جعل المختار تعليق / أو عدم جعله تعليق **Comment Selection/Uncomment Selection**

التعليقات هي سطور تقوم بإدخالها بين عبارات كودك لتوثيق تطبيقك, كل سطر يبدأ بعلامة اقتباس مفردة هو تعليق ,وهو جزء من الكود ولكن المترجم يتجاهلها . بعض الأحيان نريد أن نعطل عدد من اسطر كودنا ولكن لا نحذفها(لأننا نريد أن يكون بمقدورنا استعادتها لاحقا), التقنية البسيطة لتعطيل سطر من الكود هو جعله تعليق (وذلك بإدخال رمز التعليق أمام السطر). هذا الأمر يسمح لك بتعليق أو عدم تعليق مقاطع ضخمة من الكود بحركة واحدة(بضغطة مفردة)

القائمة تحرير < القائمة الفرعية (المساعد الفوري) Edit > IntelliSense Submenu

تقود القائمة Edit > IntelliSense إلى قائمة فرعية ذات خمس خيارات والتي تم شرحها لاحقا , و المساعد الفوري (الميزة الذكية) هي ميزة لمحرر الكود (وتطبيقات ميكروسوفت الأخرى) والتي تعرض معلومات كثيرة قدر الإمكان عندما تكون متاحة, فعندما تكتب اسم لأداة وتتبعه بالنقطة فان المساعدة الفورية(الميزة الذكية)تعرض قائمة بخصائص الأداة وطرقها لذا تستطيع اختيار الطريقة أو الخاصية المطلوبة , بدلا من تخمين اسمها ,عندما تكتب اسم الوظيفة وتفتح أقواس , فان المساعدة الفورية سوف تعرض تركيبة الوظيفة (معاملاتها النسبية its arguments) القائمة الفرعية للمساعد الفوري تتضمن الخيارات التالية:

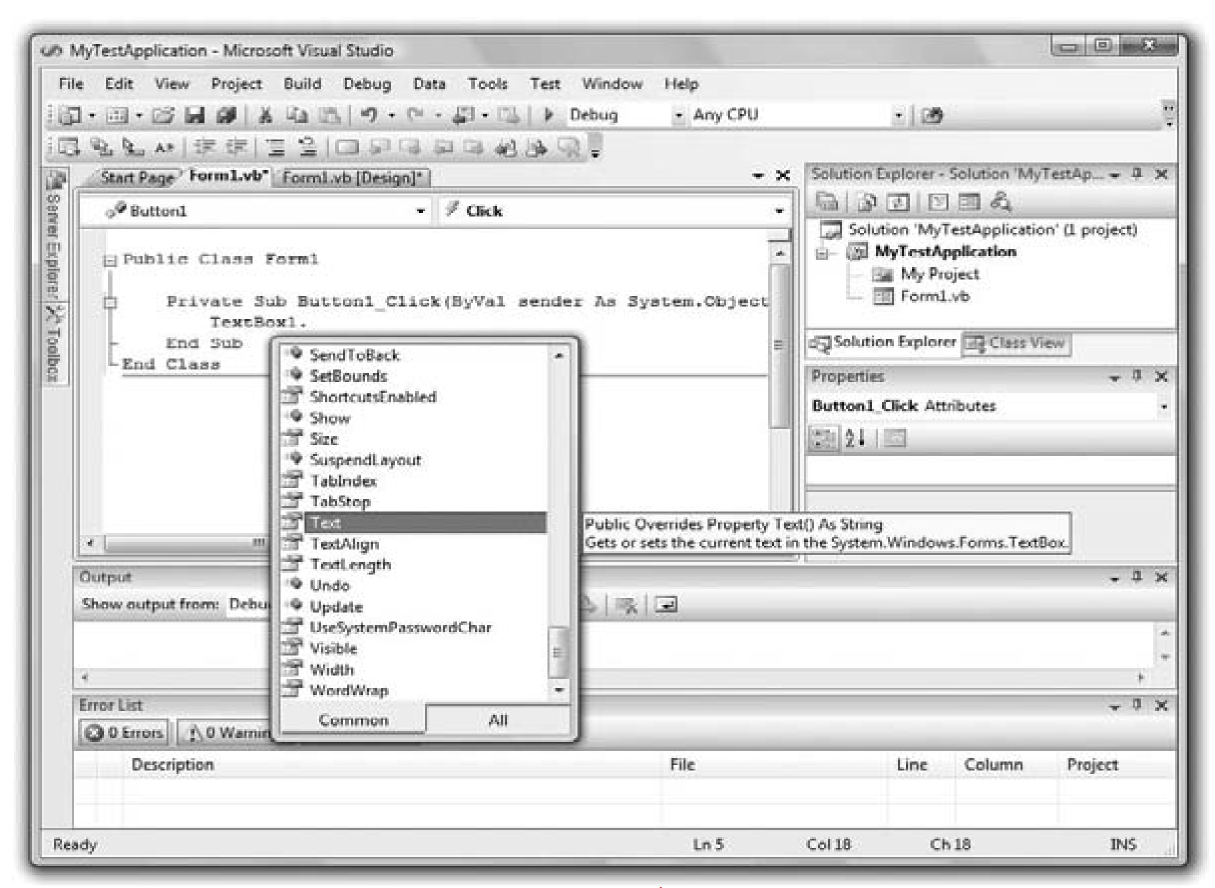
قائمة المكونات List Members

عندما يتم تفعيل هذا الخيار(on)فان محرر الكود يجدول جميع المكونات(الخصائص, الطرق , الأحداث, وقائمة المعاملات النسبية)في قائمة منسدلة وسوف تظهر هذه القائمة عندما تكتب اسم الكائن أو الأدوات متبوعة بالنقطة. ومن ثم تستطيع اختيار المكون المطلوب من القائمة بواسطة الفأرة أو لوحة المفاتيح , دعنا نقول أن نموذجك يحوي أداة مسماة TextBox1 وأنت تكتب كود لهذا الفورم فعندما تكتب اسم الأداة متبوعا بالنقطة **(.TextBox1)** فان قائمة من مكونات أداة صندوق النص سوف تظهر كما في الشكل (1,11)

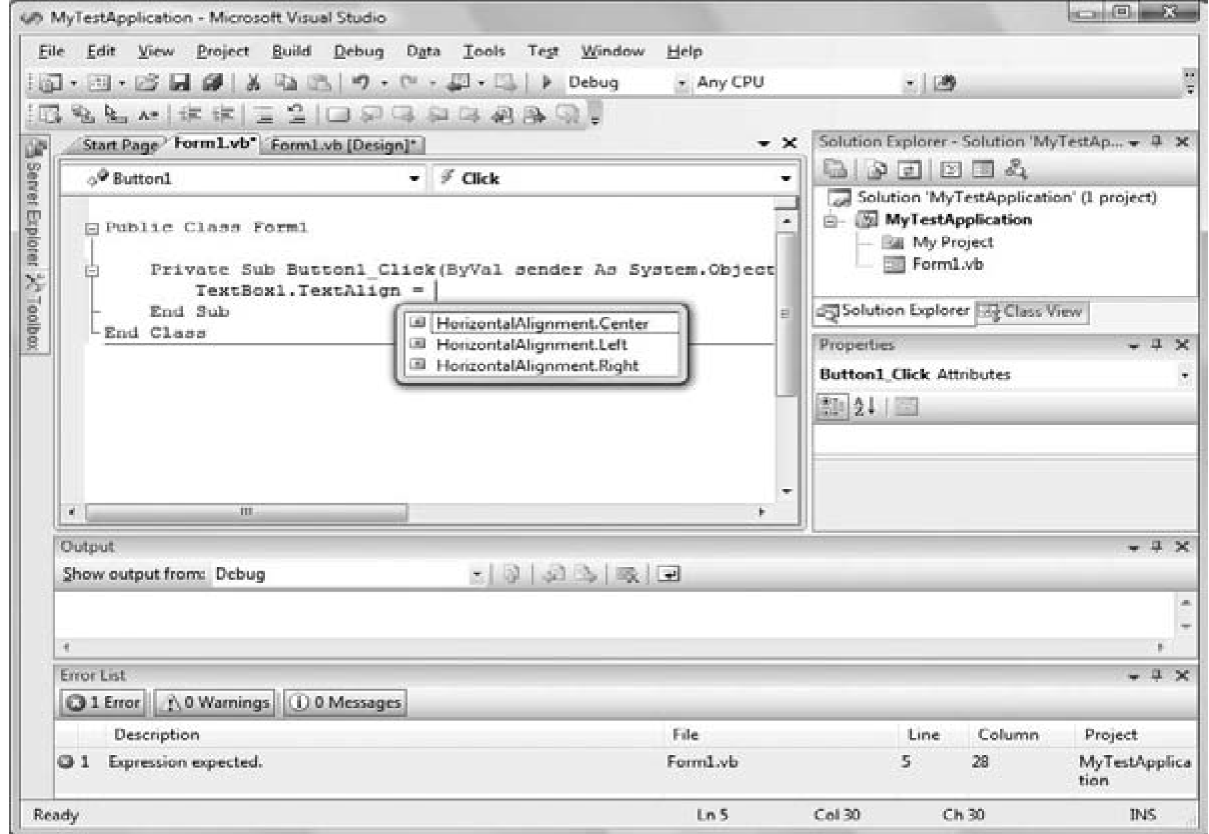
بالإضافة إلى ذلك فان وصف المكون المختار يتم عرضه في صندوق أداة فائدة ToolTip كما يمكن أن ترى في نفس الشكل اختار الخاصية text ومن ثم الإشارة (=)متبوعةً بنص موضوع ضمن علامتي اقتباس كما يلي: TextBox1.Text = "Your User Name" إذا اخترت خاصية تقبل عدد محدد من الإعدادات فانك سترى أسماء الثوابت المناسبة في القائمة المنسدلة , إذا قمت بإدخال العبارة التالية سترى الثوابت التي تستطيع إسنادها إلى الخاصية (شاهد الشكل 1,12)

TextBox1.TextAlign =

مرة أخرى تستطيع أن تختار القيمة المطلوبة بالفأرة . تبقى القائمة المنسدلة مع مكونات الأداة أو الكائن (قائمة الأعضاء) مفتوحة حتى تضغط أحد مفاتيح الانهاء (المفتاح esc أو المفتاح end) أو اختيار مكون بالضغط على مفتاح المسطرة أو enter



الشكل(1,11)عرض مكونات أداة ما في القائمة المنسدلة للمساعدة الفورية



الشكل(1,12)عرض الإعدادات الممكنة لخاصية ما في القائمة المنسدلة للمساعدة الفورية

معلومات كمية متغيرة Parameter Info

بينما أنت تحدث الكود تستطيع الانتقال بالمؤشر فوق المتغير أو الطريقة أو الخاصية وترى التصريح في صندوق أصفر بارز وتستطيع أيضا القفز إلى تعريف المتغير أو كتلة الإجراء وذلك باختيار Go To Definition من القائمة المنسدلة التي ستظهر اذا ضغطت يمين على المتغير أو اسم الطريقة في نافذة الكود

معلومات سريعه Quick Info

هذه ميزة أخرى للمساعد الفوري IntelliSense والتي تعرض معلومات الأوامر والوظائف عندما تقوم بكتابة فتح قوس (أي الرمز (من القوس) متبوعا باسم الوظيفة، مثلا المعاملات النسبيا للوظيفة ستظهر في صندوق أداة فائدة ToolTip box (صندوق أصفر أفقي) المعامل النسبي الأول يظهر بخط سميك bold وبعد إدخال القيمة للمعامل النسبي الأول فان المعامل النسبي التالي يظهر بخط سميك, إذا ما كان معامل ما يقبل عدد ثابت من الإعدادات هذه القيم ستظهر في قائمة منسدلة كما شرحت سابقاً .

إكمال الكلمات Complete Word

ميزة إتمام الكلمات تمكنك من إكمال الكلمة الحالية وذلك بالضغط على Ctrl+ spacebar مثلا إذا كتبت TEXTB ومن ثم ضغطت Ctrl+ spacebar سترى قائمة بالكلمات والتي تكون في الأغلب الأحيان مماثلة للكلمة التي تريد كتابتها مثل(TextBox, TextBox1)

إدخال قصاصة Insert Snippet

هذا الأمر يفتح نافذة إدخال قصاصةInsert Snippet في الموضع الحالي في نافذة محرر الكود, قصاصات الكود ميزة هامة للفيجوال استوديو 2008 وقد تم مناقشتها لاحقا في هذا الفصل.

تحرير>القائمة الفرعية اختصار(الخطوط العريضة) Edit > Outlining Submenu

التطبيقات العملية تحتوي كمية هائلة من الكود في عدد ضخم من معالجات الحدث وإجراءات مخصصة(إجراءات فرعية ووظائف) لتبسيط إدارة نافذة الكود فان القائمة الفرعية outlining تحتوي على أوامر والتي تقوم بتجميع ومد الإجراءات المختلفة,لنقول انك أنهيت من تحديث معالجات الحدث Click لعدة أزرار على الفورم , تستطيع تصغير reduce معالجات الحدث هذه إلى خط مفرد والذي يعرض أسماء الإجراءات و إشارة (+) أمامها, تستطيع مد قائمة الإجراء في أي وقت وذلك بالضغط على إشارة(+) التي أمام اسمه. عندما تعمل هذا إشارة(-)تظهر أمام اسم الإجراء وتستطيع ضغطها لطي collapse كتلة الإجراء مرة أخرى . القائمة الفرعية outlining تحتوي على أوامر لمعاملة الخطوط العريضة outlining للإجراءات المتنوعة,أو تعطيل (off) ألـ outlining وعرض القوائم الكاملة لجميع الإجراءات . سوف تستخدم هذه الأوامر عند كتابة تطبيقات ذات كمية ضخمة substantial amount من الكود:

مفصل توسيع الخطوط العريضة Toggle Outlining Expansion

هذا الخيار يسمح لك بتغير نسق(نمط) الخطوط العريضة للإجراء الحالي فإذا كان تعريف الإجراء procedure's definition تم طيه فانه سيوسع الكود(يمدد) والعكس بالعكس.

مفصل جمع الخطوط العريضة Toggle All Outlining

هذا الخيار مشابه للخيار السابق ولكنه يمفصل نمط الخطوط العريضة للمستند الحالي, فالفورم يتم تصغيرها إلى عبارة مفردة في كل فئة والملف المتعدد الفئات يتم تصغيره إلى سطر واحد في كل فئة.

إيقاف الخطوط العريضة Stop Outlining

هذا الخيار يعمل على تعطيل فعل الخطوط العريضة (طي/توسيع) ويضيف أمر جديد إلى القائمة الفرعية هو Start Automatic Outlining والذي تستطيع ان تختاره لتفعيل ألية الخطوط العريضة مرة أخرى.

الطي إلى التعريفات Collapse To Definitions

هذا الخيار يعمل على تصغير القوائم إلى قائمة عناوين الإجراءات.

القائمة عرض View Menu

تحوي هذه القائمة على أوامر لعرض لي شريط أدوات أو نافذة لبيئة التطوير IDE وقد رأيت مسبقاً قائمة شريط الأدوات Toolbars menu في مقطع " بداية مشروع جديد " أما نوافذ الأوامر الأخرى فإنها تقود إلى قوائم فرعية مع أسماء لبعض النوافذ القياسية , بما فيها نوافذ الإخراج والأوامرOutput and Command Windows ونافذة المخرجات هي تطبيق console حيث أن رسائل المترجم مثلا يتم عرضها في نافذة الإخراج Output window , نافذة الأمر تسمح لك بإدخال وتنفيذ عبارات عندما تعمل على تصحيح أخطاء تطبيقك,تستطيع إيقاف التطبيق وتدخل عبارات VB في نافذة الأمر

قائمة مشروع Project Menu

تحتوي هذه القائمة على أوامر لإدخال بنود إلى المشروع الحالي والبند يمكن أن يكون(نموذج form أو ملف file , مكونcomponent وحتى مشروع أخر another project)الخيار الأخير في هذه القائمة هو أمر خصائص المشروع Project Properties والذي يفتح صفحة خصائص المشروع, الأوامر Add Reference and Add Web Referenceتسمح لك بإضافة مراجع إلى مكونات الـ NET ومكونات الـ web على الترتيب(.NET components and web components).

القائمة بناء Build Menu

تحتوي على الأوامر اللازمة لبناء(ترجمة) مشروعك, الأمران الأساسيان في هذه القائمة هما بناء Build وإعادة بناء الجميع Rebuild All, الأمر بناء يترجم(يبني الملف التنفيذي executable) لكامل المشروع ولكنه لا يترجم أي من مكونات المشروع والتي لم يتم تغيرها منذ البناء الأخير, أما الأمر إعادة بناء الكل The Rebuild All يعمل بالضبط ما يعمله الأمر السابق ولكنه ينظف أي ملف موجود ويبني الحل(المشروع the solution) من جذوره.

القائمة تصحيح Debug Menu

هذه القائمة تحوي على أوامر لتشغيل أو إنهاء التطبيق بالإضافة إلى أدوات تصحيح الأخطاء الأساسية

قائمة البيانات Data Menu

تحتوي هذه القائمة على الأوامر التي ستستخدمها مع مشاريع التي تتمكن من الوصول إلى البيانات,سترى كيفية استخدام هذه النافذة في الفصلين 22و23 من هذا الكتاب.

القائمة تنسيق Format Menu

هذه القائمة تكون مرئية فقط عندما تعمل على تصميم نماذج ويندوز أو الويب (استعرض أوامرها البسيطة التي تستخدم لتنسيق الأدوات على الفورم) وهي غير مرئية عندما تعمل في محرر الكود.وأوامرها تطبق على العناصر المرئية لواجهة التطبيق.

قائمة الأدوات Tools Menu

تحتوي على قائمة بأدوات مفيدة مثل الأمر Macros(برنامجا صغيرا لعملية معينة) والتي تقود إلى قائمة فرعية بالأوامر لإنشاء macros كما تعمل المايكرو في تطبيقات الأوفيس لتبسيط العديد من المهمات, تستطيع إنشاء ماكرو لأتمتة العديد من المهمات المتكررة والتي تنجزها في بيئة التطوير IDE الأمر الأخير في هذه القائمة هو الأمر خيارات options الذي يقود الى صندوق حوار الخيارات والذي فيه تستطيع بشكل كامل أن تخصص بيئة التطوير, الأمر Choose Toolbox Items يمكنك من فتح صندوق حوار الذي بدوره يسمح لك من إضافة أدوات أخرى لصندوق الأدوات Toolbox

قائمة الويندوز Window Menu

هذه قائمة نموذجية لأي تطبيق ويندوز بالإضافة إلى قائمة فتح النوافذ تحتوي أيضا على الأمر Hide والذي يخفي جميع صناديق الأدوات تاركا كامل نافذة بيئة التطوير مكرسة devoted لمحرر الكود أو لمصمم النماذج ,لا تختفي صناديق الأدوات بشكل كامل بل تنكمش retracted وتستطيع رؤية مقابضهم على الحافة اليمنى واليسرى لبيئة التطوير.لتوسيع صندوق الأدوات فقط قم بأرجحة المؤشر على المقبض الموافق corresponding tab.

قائمة المساعدة Help Menu

MSDN

نافذة صندوق الأدوات Toolbox Window

تحتوي على جميع الأدوات التي تستطيع استخدامها لبناء واجهة تطبيقك application's interface وهذه النافذة في العادة تنكمش retracted وعليك تحريك المؤشر عليها لعرض صندوق الأدوات والأدوات تكون منظمة في مقابض متنوعة لذلك استكشف هذه الأنواع لتتعرف على هذه الأدوات ووظائفهم في الفصول القليلة الأولى سنعمل مع الأدوات في التنظيم (الأدوات المشتركة و القوائم , الأشرطة)(Common Controls and Menus & Toolbars tabs)

نافذة مستكشف الحل Solution Explorer Window

تحتوي على قائمة بعناصر الحل الحالي والحل solution يمكن أن يحتوي على العديد من المشاريع وكل مشروع يمكن أن يحتوي على العديد من البنود , مستكشف الحل يعرض قائمة شجرية hierarchical list بجميع المركبات المنظمة بواسطة المشروع ,تستطيع بضغط يمين على مكون للمشروع وتختار خصائصه من القائمة المنسدلة لترى خصائص المكون المختار في نافذة الخصائص,إذا اخترت مشروع ,سترى صندوق حوار خصائص المشروع, إذا كان الحل يحوي العديد من المشاريع تستطيع ضغط يمين على المشروع الذي تريد أن يصبح نموذج البداية (الإقلاع) وتختار Set As Startup Project تستطيع أيضا إضافة بنود إلى المشروع بواسطة الأمرAdd Item من القائمة المنسدلة أو إزالة احد المكونات للمشروع بالأمر Exclude From Project هذا الأمر يزيل المكون المختار من المشروع ,ولكن لا يؤثر على مكونات الملف في القرص , الأمر delete يزيل المكون المختار من المشروع وأيضا يحذف ملف المكون من القرص .

نافذة الخصائص Properties Window

هذه النافذة (تعرف أيضا بمستعرض الخصائص Properties Browser) تعرض جميع الخصائص للمكون المختار وإعداداته في كل مرة تضع فيها أداة على الفورم تنتقل إلى هذه النافذة لتعدل مظهر الأداة ,ولقد رأيت مسبقا كيفية معالجة خصائص أداة ما من خلال نافذة الخصائص ,العديد من الخصائص تسند إلى قيمة مفردة مثل عدد أو نص. إذا كانت الإعدادات الممكنة لخاصية ما قليلة نسبيا relatively few يتم عرضها كثوابت ذات معنى هام في قائمة السياق ,بعض الخصائص يتم إعدادها من خلال واجهات محكمة فمثلا خاصية اللون Color properties يتم إعدادها من ضمن صندوق حوار اللون Color dialog box والذي يتم عرضه من بشكل كامل في نافذة الخصائص, خصائص الخط يتم إعدادها من خلال صندوق حوار الخط المألوف, المجمعات يتم إعدادها في صندوق حوار محرر المجمع Collection Editor Dialog box والتي تستطيع فيها إدخال نص لكل بند للمجمع' كما فعلت مع بنود أداة ComboBox مسبقا في هذا الفصل ,إذا كانت نافذة الخصائص غير ظاهرة أو قمت بإغلاقها, تستطيع إما أن تختار عرض View> نافذة الخصائص Properties Window , أو أن تضغط يمين على أي اداة على الفورم وتختار خصائص properties أو تستطيع بسهولة الضغط على F4 لإحضار نافذة الخصائص ,في بعض الحالات عندما تغطي أداة ما بشكل كامل أداة أخرى ولن تكون قادرا على اختيار هذه الأداة المخفية تحت الأداة التي تخطتها overlap وعرض خصائصها في هذه الحالة تستطيع اختيار الأداة المطلوبة في الصندوق المركب ComboBox الذي أعلى نافذة الخصائص ,هذا الصندوق يحتوي على أسماء جميع الأدوات التي توضع على الفورم,وبالتالي تستطيع أداة على الفورم باختيار اسمها في هذا الصندوق .

نافذة الإخراج Output Window

هذه النافذة هي المكان الذي فيه العديد من الأدوات ومن ضمنها المترجم ترسل مخرجاتها إليها، في كل مرة يبدأ التطبيق فان سلسلة من الرسائل يتم عرضها في نافذة المخرجات وهذه الرسائل يتم إنتاجها بواسطة المترجم, ولا تحتاج الى فهمها في الوقت الحالي ,إذا كانت نافذة المخرجات غير مرئية, اختار عرض View > نوافذ أخرى > نافذة المخرجات Output من القائمة.

نوافذ الأمر و المباشرة(الفورية) Command and Immediate Windows

بينما تجرب البرنامج,تستطيع قطع تنفيذه وذلك بإدخال ما يدعى بنقطة الإيقاف.breakpoint عندما يصل تنفيذ التطبيق إلى نقطة الإيقاف فان تنفيذ البرنامج يتوقف مؤقتا suspended , وتستطيع تنفيذ عبارة ما في نافذة Immediate فأي عبارة تظهر في كود فيجوال بيسك تستطيع أيضا أن تنفذها في نافذة Immediate لتقيم تعبير ما ,أدخل علامة استفهام متبوعة بالتعبير الذي تريد أن تقيمه كما في المثال التالي حيث أن result هو متغير في البرنامج الذي قمت بقطع تنفيذه interrupted

```
? Math.Log (35)
? "The answer is " & result.ToString
```

تستطيع أيضا أن ترسل المخرجات لهذه النافذة من ضمن كودك بالطريقة Debug. Write والطريقة Debug.WriteLine . بشكل فعلي تستخدم هذه التقنية لتصحيح الأخطاء بشكل واسع وذلك بطباعة قيم متغير تم إنشاءه قبل الدخول حيز الخطر للكود . يوجد أدوات محكمة أخرى لمساعدتك في تصحيح أخطاء تطبيقك, ولكن طباعة القليل من القيم في نافذة Immediate هو تدريب قديم a time-honored practice (متمتع بقداسة القدم) في البرمجة في VB وفي العديد من أمثلة هذا الكتاب وخاصة في الفصول القليلة الأولى سأستخدم العبارة WriteLine لطباعة شيء ما في نافذة immediate لتوضيح استخدام الوظيفة DateDiff() function مثلا سأستخدم عبارة مثل التالية:

```
Debug.WriteLine(DateDiff(DateInterval.Day, #3/9/2007#, #5/15/2008#))
```

عندما يتم تنفذ هذه العبارة فان القيمة 433 سوف تظهر في نافذة Immediate هذه العبارة توضح تركيب الوظيفة (فرق التاريخ DateDiff() function) والتي تعود بالفرق بين تاريخين بالأيام .وإرسال بعض المخرجات إلى النافذة Immediate لاختبار وظيفة أو عرض نتيجة حسابات مباشرة هو تدريب معروف ,للحصول على فكرة عن مهمة(دلالة) functionality النافذة Immediate window أقفل عائدا إلى تطبيق المثال الأول وأدخل عبارة Stop بعد عبارة End If في handler Click event معالج حدث ضغط الزر نفذ البرنامج ,واختار اللغة ,واضغط الزر على الفورم, بعد عرض صندوق الرسالة ,سيصل التطبيق إلى عبارة الإيقاف وسيتم قطع تنفيذه

وسترى نافذة Immediate أسفل بيئة التطوير IDE إذا كانت غير مرئية افتح النافذة Debug menu واختار Windows > Immediate في نافذة Immediate ادخل العبارة التالية :

```
? ComboBox1.Items.Count
```

ومن ثم اضغط enter ليتم تنفيذ العبارة السابقة,لاحظ أن المساعد الفوري IntelliSense موجود بينما تقوم بالكتابة في النافذة Immediate .يطبع التعبير السابق عدد بنود الأداة ComboBox control (لا تقلق بخصوص العدد الهائل من الخصائص للأداة وللطريقة التي أحضرتهم بها هنا,لقد تم مناقشتهم بالتفصيل في الفصل السادس) حالما تضغط انتر فان القيمة 5 ستطبع في الخط التالي . تستطيع أيضا معالجة الأدوات على الفورم من ضمن نافذة Immediate window ادخل العبارة التالية واضغط إنتر لتنفيذها:

```
ComboBox1.SelectedIndex = 4
```

البنود الخمس التي على الأداة سيتم اختيارها (فهرسة البنود يبدأ بالصفر0) ومهما يكن, لا تستطيع أن ترى تأثيرات تغيراتك لان التطبيق غير منفذ اضغط F5 لمتابعة تنفيذ التطبيق سترى أن البند Cobol هو الآن في وضع الاختيار في الأداة .ComboBox control نافذة Immediate ممكنة فقط بينما يكون تنفيذ التطبيق تم إيقافه مؤقتا is suspended لتتابع التجريب عليه ,اضغط الزر الذي على الفورم للتحقق من اختيارك choice عندما يتم تنفيذ عبارة الإيقاف مرة أخرى ,ستيتم أعادتك إلى النافذة Immediate window.

نافذة الأمر Command window غير مشابهة لنافذة Immediate فنافذة الأمر متوفرة في وقت التصميم (available at design time)نافذة الأمر تسمح لك من الوصول إلى جميع أوامر الفيجوال أستوديو وذلك بكتابة أسمائها في هذه النافذة, إذا أدخلت النص **Edit** متبوعا بالنقطة,سترى قائمة بجميع أوامر القائمة Edit,متضمنة تلك التي تكون غير مرئية في الوقت الحالي,وتستطيع تنفيذ أي من هذه الأوامر وتمرير معاملات نسبية لها.على سبيل المثال إذا أدخلت **Edit. Find "Margin"** في نافذة الأمر ومن ثم ضغطت إنتر فانه سيتم إيجاد

النسخة الأولى من النص Margin في نافذة الكود المفتوح, لتشغيل التطبيق تستطيع كتابة Debug. Start تستطيع إضافة مشروع جديد للحل الحالي بالأمر AddProj وهكذا. معظم المطورين وحتى بالكاد hardly يستخدمون هذه النافذة وقت التصميم أو وقت التنفيذ وتصحيح الأخطاء in designing or debugging applications

نافذة قائمة الخطأ Error List Window

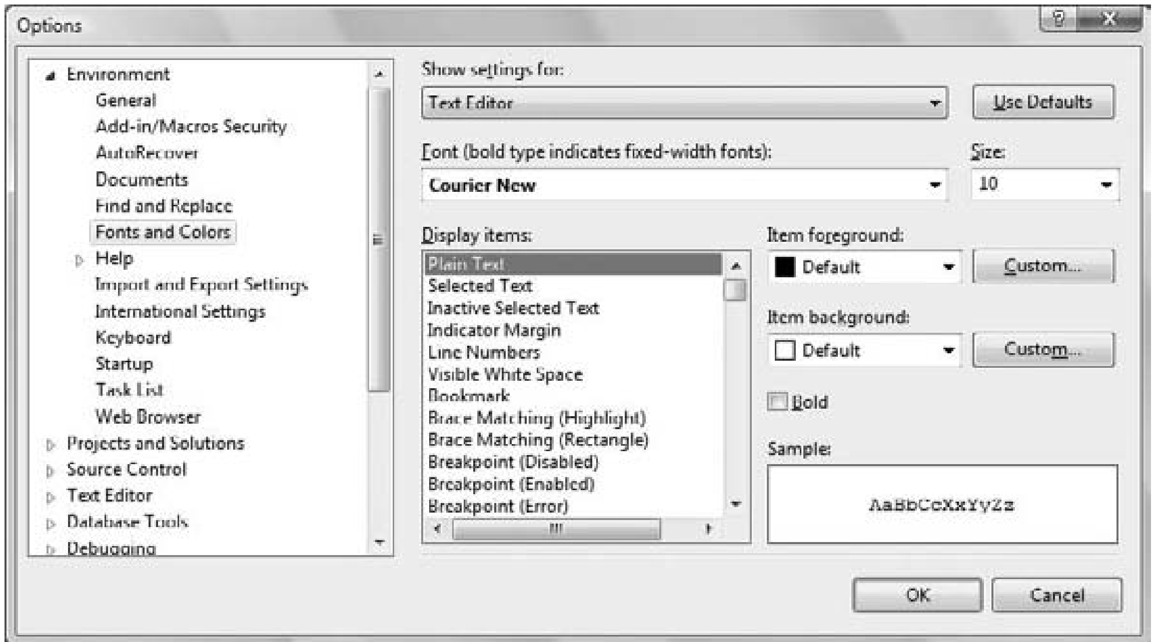
هذه النافذة يتم تعبئتها بواسطة المترجم برسائل الخطأ إذا لم تنجح عملية ترجمة الكود. تستطيع ضغط مزدوج على رسالة خطأ ما في هذه النافذة وستأخذك بيئة التطوير IDE إلى السطر الذي فيه عبارة الخطأ والتي ستقوم بإصلاحها, بدل اسم الوظيفة MsgBox() function إلى **MssgBox().** حالما تغادر السطر الذي فيه الخطأ فانه سيتم وضع خط أحمر متعرج تحت اسم الوظيفة wiggly red line الوصف التالي للخطأ سيظهر في نافذة قائمة الخطأ

Name ' MssgBox ' is not declared

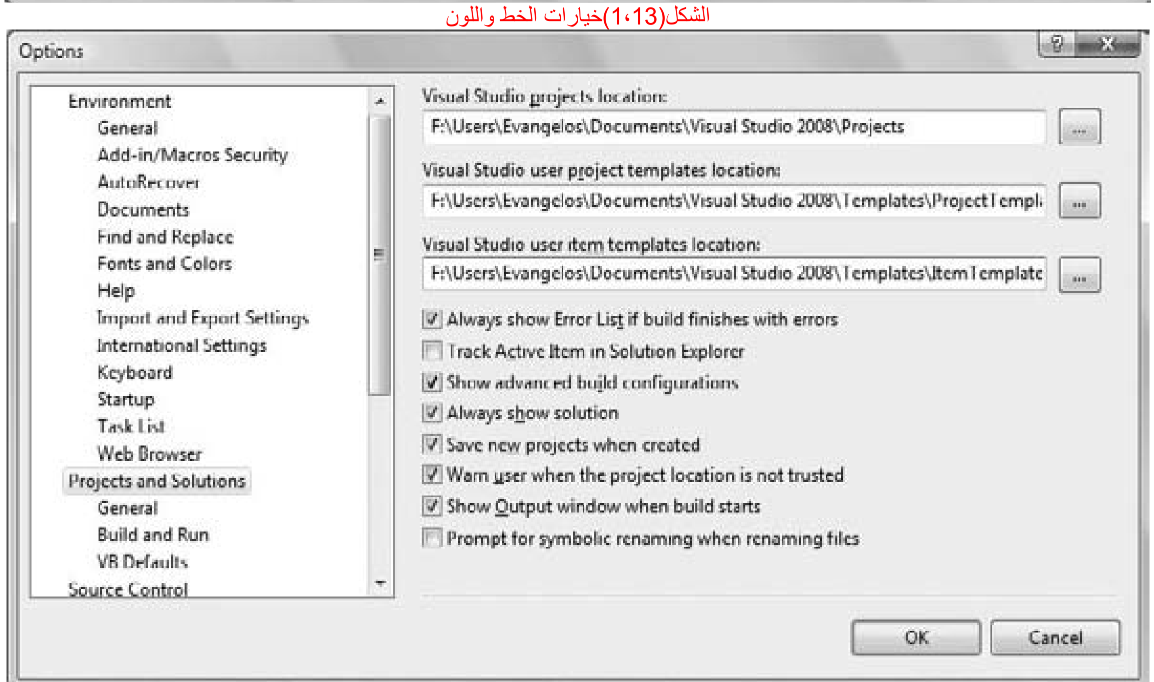
Setting Environment Options

خيارات إعداد بيئة التطور Setting Environment Options

بيئة التطوير للفيجوال أستوديو قابلة للتخصيص بشكل عالي. سأريك كيف تقوم بتغير الإعدادات الافتراضية لبيئة التطوير , افتح القائمة أدوات Tools menu واختار Options(البند الأخير في القائمة) وبالتالي يظهر صندوق حوار الخيارات Options والذي به تستطيع إعداد جميع الخيارات تبعا للبيئة، الشكل(1،13) يظهر خيارات خطوط بنود بيئة التطوير المتنوعة هنا, تستطيع إعداد الخط لمحرر النص,صناديق الحوار,صناديق الأدوات,وهكذا, اختار بند ما في الشجرة في قائمة اللوحة اليسارية ومن ثم اسند الخط لهذا البند في الصندوق الأسفل. الشكل (1،14) يظهر خيارات المشروعات والحلول بدل الصندوق العلوي على المكان الافتراضي للمشروع جديد, حفظ المشروعات الجديدة عندما يكون صندوق الاختيار Created check box يحدد فيما إذا المحرر سيقوم بإنشاء مجلد في مجلد المؤقت Temp folder . المشاريع في المجلد المؤقت ستتم إزالته عندما تشغل(run the Disk Cleanup utility أداة منظف القرص لتوفير مساحة أكبر على سواقتك . بشكل افتراضي VS يحفظ التغيرات للمشروع الحالي في كل مرة تضغط F5 تستطيع تغير هذا السلوك بواسطة إعداد الخيار Before Building(قبل التنفيذ أو البناء)في الصفحة Build And Run page (بناء وتنفيذ)تحت الفرع Project And Solutions branch فإذا ما غيرت هذه الإعدادات يتوجب عليك حفظ مشروعك من وقت لأخر من القائمة ملف File <الأمر حفظ الكل (Save All) إذا أردت تغير الإعدادات الافتراضية لبيئة التطوير فهذا هو المكان المناسب لذلك.



الشكل(1،13)خيارات الخط واللون



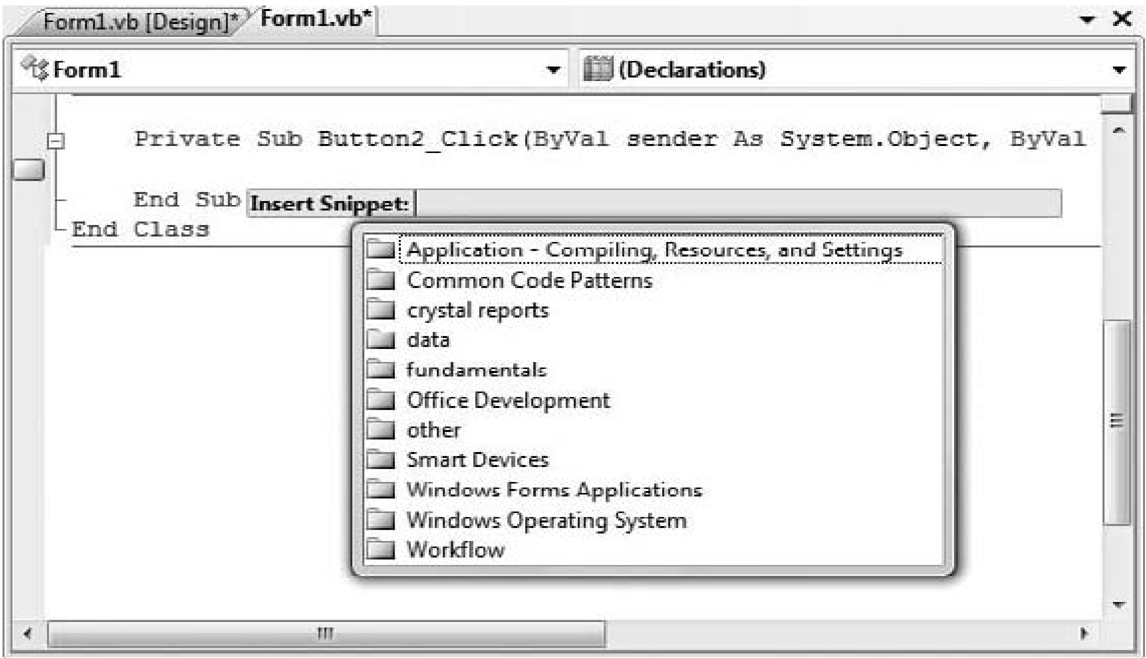
الشكل(1،14)خيارات المشاريع والحلول

بناء تطبيقات console

استخدام قصاصات الكود Using Code Snippets

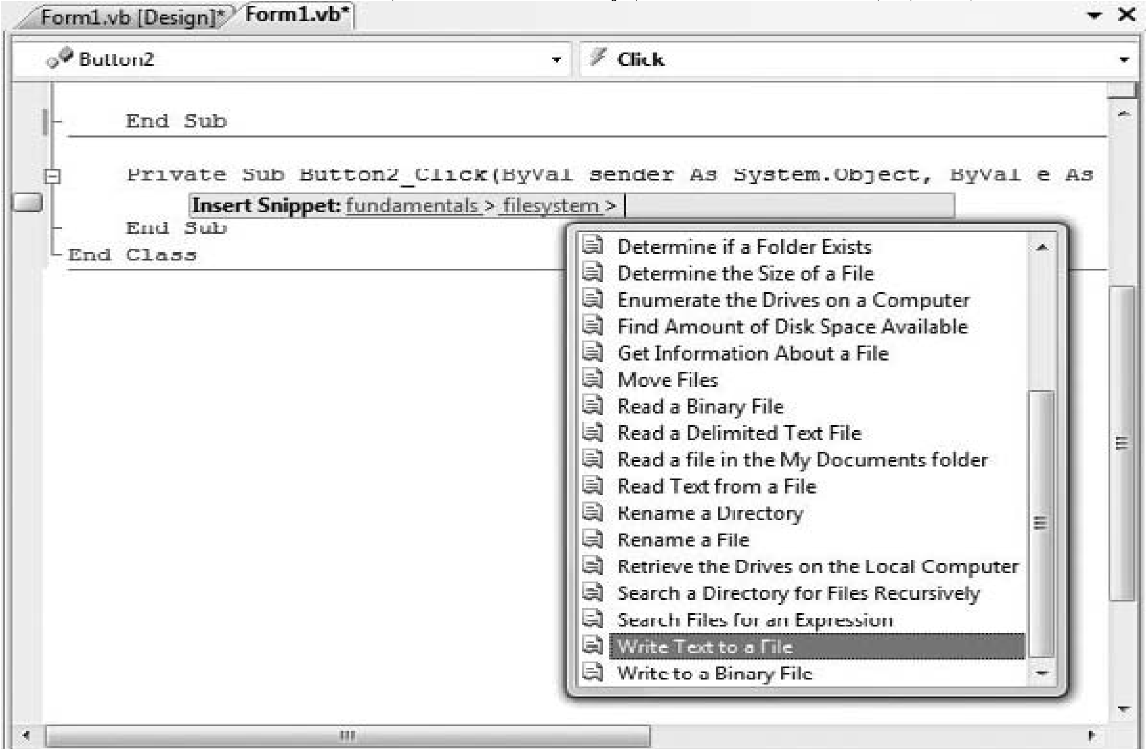
تأتي الفيجوال أستوديو2008 مع الكثير من قصاصات الكود المعرفة مسبقا لأفعال مختارة، وتستطيع إدخال هذه القصاصات في كودك إذا احتجت لذلك,لنقول انك تريد إدخال عبارات لكتابة بعض النصوص إلى ملف ,ولكن ليس لديك فكرة عن كيفية الوصول إلى الملف ,قم بإنشاء سطر فارغ في الكود (اضغط المفتاح انتر مرتين في نهاية سطر الكود) ومن ثم افتح قائمة تحرير edit واختار IntelliSense > Insert Snippet) أو اضغط يمين في السطر الفارغ الذي أنشأته في نافذة محرر الكود ومن القائمة المنسدلة اختارInsert Snippet)سترى على الشاشة قائمة بالقصاصات منظمة في ملف تبعا لوظائفهاfunction) كما في الشكل (1،17)اختار المجلد fundamentals folder والذي سيعرض قائمة خيارات أخرى :
collections and arrays, data types, file system, and math اضغط مزدوج على البند file system لترع قائمة بالمهمات المشتركة المرتبطة بالملف common file-related tasks حدد البند Write Text To A File من القائمة واضغط مزدوج عليه لإدخال القصاصة المناسبة في الموقع الحالي من نافذة الكود.
العبارة التالية من قصاصة الكود سيتم إدخالها في الكود

```
My.Computer.FileSystem.WriteAllText("C:\Test.txt", "Text", True)
```



الشكل(1،16) قصاصات الكود منظمة تبعا لوظائفها

لكتابة بعض النصوص إلى الملف عليك استدعاء الطريقة WriteAllText method للكائن FileSystem . My. Computer تستطيع استبدال النصوص الظاهرة في القصاصة بقيم فعلية, فالنص الأول هو اسم الملف "C:\Test.txt")أما النص الثاني"Text" فهو النص الذي ستتم كتابته إلى الملف والمعامل النسبي الأخير argument للطريقة يحدد فيما إذا سيتم تزيل (إلحاق) النص إلى الملف (إذا كانت if False) أو سيتم الكتابة فوق أي نص موجود إذا كانت (if True) . القصاصات تريك العبارات الأساسية من أجل انجاز المهمات المشتركة العامة common task وتستطيع تحرير وتحديث الكود المدخل بواسطة الفيجوال أستوديو اذا احتاج الأمر لذلك. التطبيق الحقيقي يمكن أن يطلب من المستخدم اسم الملف بواسطة صندوق الحوار المعروف للملف ومن ثم يستخدم اسم الملف المحدد بواسطة المستخدم في صندوق الحوار,بدلا من اسم الملف الجامد المزود بواسطة الكود.



الشكل(1،17) اختيار قصاصة الكود لإدخالها في كودك

عندما تبرمج ,عليك محاولة استكشاف فيما إذا يوجد قصاصة للمهمة التي تبرمجها معدة للاستعمال, في بعض الأحيان تستطيع استخدام القصاصة بدون حتى معرفة كيف تعمل ,على الرغم من أن القصاصات تبسيط الأمور ,إلا أنها لن تساعدك على فهم منصة العمل (Framework) والتي ستناقش في التفصيل في هذا الكتاب.

استخدام العبارة MyObject

من المحتمل انك لاحظت أن قصاصة الكود في الفيجوال أستوديو تستخدم كينونة تدعى My والتي هي كائن خاص تم تقديمه بواسطة VB 2005 لتبسيط العديد من مهام البرمجة , وكما رأيت الكائن My(خاصتي)يسمح لك بكتابة بعض النصوص إلى ملف ما بعبارة مفردة ,بالطريقة WriteAllText method إذا كنت تعرف الإصدارات السابقة من الفيجوال بيسك فأنت

تعلم انه عليك فتح ملف أولا ومن ثم كتابة بعض النص إليه , وأخيرا إغلاق الملف ,الكائن My يسمح لك بإنجاز كل هذه العمليات بعبارة وحيدة, كما رأيت في المثال السابق,ومثال أخر : الطريقة Play method التي تسمح لك بتشغيل ملف ويف WAV file من ضمن كودك:

```
My.Computer.Audio.Play("C:\Sounds\CountDown.wav")
```

تستطيع استخدام التعبير التالي لتشغيل ملف صوت النظام:

```
My.Computer.Audio.PlaySystemSound(System.Media.SystemSounds.Exclamation)
```

فالطريقة التي تشغل plays back الصوت هي الطريقة Play method والطريقة التي تكتب نص إلى ملف هي WriteAllText method ولكن مهما يكن لا تستطيع استدعائهم مباشرة بالكائن My object فهم ليست بطرق له , فإذا كانت كذلك عليك التنقيب بعمق لاكتشاف الطريقة التي تحتاجها ,الكائن My object يعرض ستة مكونات والتي تحتوي بدورها على مكوناتها الخاصة إليك شرح المكونات الأساسية للكائن My والوظيفة التي تتوقع إيجادها في كل مكون :

My.Application

مكون التطبيق يزود بالمعلومات حول التطبيق الحالي فالخاصية CommandLineArgs property لـ My. Application تعود بمجمع نصي والذي هو المعاملات النسبية الممررة إلى التطبيق عندما تم تشغيله ، فتطبيقات ويندوز النموذجية لا يتم استدعائها بالخاصية command-line arguments ولكن من المحتمل تشغيل تطبيق ما وتمرير اسم الملف كمعامل نسبي للتطبيق (مثال المستند الذي سيتم فتحه بواسطة التطبيق) . الخاصية Info property هي كائن يعرض الخاصيات مثل DirectoryPath(مجلد التطبيق الافتراضي), اسم المنتج Product Name ,الإصدارVersion وهكذا .

Computer

هذا المكون للكائن My يعرض الكثير من المهمات بواسطة عدد من الخصائص , والعديد منها هي كائنات, فالمركب My.Computer.Audio يسمح لك بتشغيل الأصوات والمكون My.Computer.Clipboard يمكنك من الوصول إلى الحافظة لاستكشاف فيما إذا كانت الحافظة تحتوي على نوع معين من البيانات , استخدم الطرق ,ContainsText ContainsImage, وContainsAudio ContainsData, and , واستخدم الطرق GetText, GetImage, GetData, and GetAudioStream لاستخلاص محتوى الحافظة . افترض أن لديك نموذج مع أداة صندوق نص وأداة صندوق صورةPictureBox تستطيع استخلاص بيانات النص أو الصورة من الحافظة وعرضها على الأداة المناسبة وذلك باستخدام التعبير التالي:

```
If My.Computer.Clipboard.ContainsImage Then
        PictureBox1.Image = My.Computer.Clipboard.GetImage
    End If
    If My.Computer.Clipboard.ContainsText Then
        TextBox2.Text = My.Computer.Clipboard.GetText
End If
```

ربما لاحظت أن استخدام الكائن Myفي كودك يحتاج إلى كتابة عبارات طويلة . تستطيع تقصير هذه العبارات بشكل جوهري باستخدام العبارة With كما في المثال التالي :

```
With My.Computer.Clipboard
        If .ContainsImage Then
            PictureBox1.Image = .GetImage
        End If
        If .ContainsText Then
            TextBox2.Text = .GetText
        End If
End With
```

فعندما تنفذ العديد من العبارات على نفس الكائن تستطيع تخصيص الكائن بواسطة العبارة With واستدعاء طرقها في بلوك العبارة With وذلك بتخصيص اسم الطريقة مسبوق النقطة كما في المثال السابق ,وعبارة With تكون متبوعة باسم الكائن والذي عليه تطبق كل الطرق اللاحقة , ويتم إنهائها بالعبارة End With

توجد خاصية أخرى لمكون My.Computer وهي الكائن FileSystem والذي يعرض جميع الطرق التي تحتاجها للوصول إلى الملفات والمجلدات. فإذا أدخلت: My.Computer.FileSystem متبوعا بنقطة سترى جميع الطرق المعروضة بواسطة المكون FileSystem ومن بينها ستجد DeleteFile و DeleteDirectory , RenameFile وRenameDirectory, WriteAllText, ReadAllText , اختار الطريقة ومن ثم اكتب فتح قوس سترى تركيب الطريقة في أداة الفائدة ToolTip فتركيب الطريقة CopyFile هو كما يلي :

```
My.Computer.FileSystem.CopyFile(sourceFileName As String, destinationFileName As _    String,overwrite As Boolean)
```

المعامل النسبي overwrite argument يحدد فيما إذا الطريقة ستكتب فوق الملف المقصود إذا كان موجود, الإصدار الثالث من الطريقة يقبل معامل نسبي ثالث والذي يحدد فيما إذا آلية النسخ العادي usual copy animation سيتم عرضها بنفس تنسيق الملف الذي تم نسخه as the file is being copied,الإصدارات المتنوعة لنفس الطريقة تختلف في العدد و/أو النوع في معاملاتها ,وهذا ما يدعى (بنماذج إعادة تعريف overloaded forms) الطريقة, فبدلا من استخدام أسماء طرق متعددة لنفس العملية الأساسية , نماذج إعادة التعريف للطريقة تسمح لك باستدعاء نفس اسم الطريقة وتعديل سلوكها بواسطة تحديد معاملات نسبية مختلفة لها .

النماذج Forms

هذا المكون يسمح لك بالوصول لنماذج (forms) التطبيق الحالي ,وتستطيع أيضا الوصول إلى نماذج التطبيق الحالي بواسطة الاسم, لذا فان مكون الفورم ليس المكون المفيد كثيرا.

الإعدادات Settings

هذا المكون يسمح لك بالوصول إلى إعدادات التطبيق ,وهذه الإعدادات تطبق على كامل التطبيق وتخزن في ملف تركيبي(إعداد) an XML configuration file من نوع xml يتم إنشاء الإعدادات من ضمن الفيجوال أستوديو وأنت تستخدم ا لمكون Settings لقراءتها.

User

هذا المكون يعود بالمعلومات حول المستخدم الحالي, الخاصية الأكثر أهمية لمكون المستخدم(User component) هي الخاصية CurrentPrincipal property والتي هي كائن يمثل وثائق(اعتمادات) credentials المستخدم الحالي.

Web Services

يمثل خدمات الويب المقدمة بواسطة التطبيق الحالي.

إن الكائن يمنح المبتدئين قوة غير مسبوقة ويسمح لك بانجاز المهمات التي تتطلب كود ضخم إذا ما عالجتها باستخدام إصدارات سابقة من اللغة,هذا بغض النظر عن عملية البحث والتي ستستغرقها في إيجاد الطرق المناسبة في منصة عمل Framework. تستطيع استكشاف الكائن My بنفسك وتستخدمه كلما احتجت لذلك, إن My ليس البديل substitute عن تعلم اللغة ومنصة العمل Framework فهي تستطيع مساعدتك بشكل أولي , ولكن لا تستطيع الذهاب بعيدا بدون تعلم طرق Framework لمعالجة الملفات أو أية ميزة أخرى , دعنا نقول انك تريد إيجاد جميع الملفات من نوع معين في مجلد ما, ومن ضمنها المجلدات الفرعية .عملية مسح المجلد ومجلداته الفرعية عند أي مستوى هي مهمة كاملة بحد ذاتها,(ستجد الكود في الفصل 15) تستطيع فعل نفس الشيء بعبارة وحيدة باستخدام الكائن My :

```
Dim files As ReadOnlyCollection(Of String)
files = My.Computer.FileSystem.GetFiles("D:\Data", True, "*.txt")
```

الطريقة GetFiles تملأ مجمع الملف files collection بأسماء المسارات pathnames للملفات النصية في المجلد D:\Data ومجلداته الفرعية, ولكن مهما يكن فهي لن تساعدك إذا أردت معالجة كل ملف في مكانه, وأكثر من ذلك هذه الطريقةGetFiles هي عملية متزامنة(تحدث في نفس الوقت synchronous)فإذا كان المجل يحوي عدة مجلدات فرعية مع العديد من الملفات ستثبط(block) الواجهة حتى يتم استخلاص كافة الملفات, في الفصل 15 سترى الكود الذي يستخلص أسماء الملفات ويضيفها إلى أداة كلما تقدمنا إلى الأمام .

إذا كان لديك معرفة مسبقة بالفيجوال بيسك ربما تظن أن الكائن my هو مساعدة للمبتدئين تماما أو الغير مبرمجين, وهذا ليس صحيحا, فالفيجوال هي تقريبا لغة خصبة (ذات قابلية للإنتاج productivity) والكائن My يساعدك لأن تكون أكثر قابلية للإنتاج في مهماتك اليومية , بغض النظر عن معرفتك باللغة أو مهارات البرمجة.فإذا كان بإمكانك استخدام My لحفظ العديد من العبارات(أو العديد من حزم العبارات) افعل ذلك, فلا يوجد حظر penalty على استخدام الكائن My لأن المترجم يستبدل طرق هذا الكائن بالطريقة المكافئة المدعوة في (منصة العمل Framework

الفصل الثاني (2)

المتغيرات وأنواع البيانات Variables and Data Types

هذا الفصل والذي يليه يناقش أساسيات أية لغة برمجة (المتغيرات وأنواع البيانات) فالمتغيرات تخزن البيانات التي يتم معالجتها بالعبارات ،والبرنامج هو قائمة من العبارات التي تعالج المتغيرات,وحتى لكتابة تطبيقات بسيطة تحتاج إلى فهم جوهري لبعض المواضيع القاعدية ,مثل أنواع البيانات(نوع أو صنف البيانات التي تستطيع تخزينها في المتغير) ومجال (scope) وعمر المتغير(lifetime) وكيفية كتابة الإجراءات وتمرير المعاملات النسبية لها, في هذا الفصل ستستكشف الأنواع الأساسية للفيجوال أستوديو وفي الفصل اللاحق ستتعلم الإجراءات وعبارات تحكم التدفق (الانسياب)

في هذا الفصل ستتعلم كيفية عمل التالي:

- ✓ التصريح (الإعلان) واستخدام المتغيرات
- ✓ استخدام أنواع البيانات الأصلية native
- ✓ إنشاء أنواع بيانات مخصصة
- ✓ استخدام المصفوفات arrays

ملاحظة: قبل الخوض في غمار المتغيرات تسطيع تنفيذ أي من الاكواد في هذا الفصل والفصول اللاحقة في حدث ضغط الزر بوضع زر على الفورم وفتح محرر الكود بالضغط على هذا الزر مرتين وتستطيع ارسال ناتج عملية ترجمة الكود الى النافذة المباشرة immediate window وذلك من خلال كتابة Debug.WriteLine أو باستخدام صندوق الرسالة MsgBox

المتغيرات Variables

للمتغير اسم وقيمة فالمتغير Username مثلا يمكن أن يكون لديه القيمة (أحمد) والمتغير Discount يمكن أن تكون قيمته 0.35 إن Username و Discount هي أسماء المتغيرات و ((أحمد) و0.35) هي قيم المتغيرات (أحمد) هو نص (والذي هو نص أو قيمة ترقيم هجائي(حرفي) (alphanumeric) والقيمة 0.35 هي قيمة رقمية ,وعندما تكون قيمة المتغير نصية يجب أن تضمن ضمن علامتي اقتباس(" ") في كودك , بإمكانك أن تشير إلى قيمة المتغير بواسطة اسمه , هذا بالإضافة إلى اسم المتغير وقيمته ,المتغيرات لديها نوع البيانات ,التي تحدد ما هو نوع القيم التي نستطيع أن نخزنها في المتغير , تدعم الفيجوال بيسك 2008 العديد من أنواع البيانات, التي هي في الواقع لغة التنفيذ المشتركة Common Language Runtime (CLR) التي تدعم أنواع البيانات, فهذه البيانات مشتركة لجميع اللغات وليس فقط للفيجوال بيسك . إن نوع البيانات لمتغير ما يتم تحديدها عندما يتم الإعلان عن المتغير , وسوف تصرح دائما عن المتغيرات قبل استخدامها ,للتصريح عن متغير ادخل العبارة Dim متبوعة باسم المتغير والكلمة المحجوزة As ومن ثم نوع المتغير .

```
Dim Amount As Decimal
```

إن Decimal هو نوع بيانات رقمية والتي تستطيع أن تخزن كلا من قيم الأعداد الصحيحة integer والغير صحيحة noninteger فمثلا العبارة التالية تحسب وتعرض الحسم للكمية $24,500 افتح مشروع جديد و أضف إلى الفورم فقط زر button واضغط مزدوج على الزر ليتم فتح محرر الكود والذي ينقلك مباشرة إلى حدث معالجة نقر الزر وفيه اكتب العبارات التالية كما هو مبين أدناه :

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)_          Handles Button1.Click
    Dim Amount As Decimal
    Dim Discount As Decimal
    Dim DiscountedAmount As Decimal
    Amount = 24500
    Discount = 0.35
    DiscountedAmount = Amount * (1 - Discount)
    MsgBox("Your price is $" & DiscountedAmount.ToString)
End Sub
```

ربما يكون المترجم قد وضع خط متعرج احمر تحت العبارة التي تسند القيمة 0.35 إلى المتغيرDiscount وولد رسالة خطأ لعرض رسالة الخطأ حرك المؤشر فوق المقطع الذي تحته خط , هذا الخطأ سيحدث فقط إذا كان خيار التدقيق مفعّل(option strict is on) ولكن بشكل افتراضي فان هذا الخيار يكون معطل(off), المترجم يعامل treats أية قيمة رقمية مع جزء كسري على أنها قيمة مزدوجة(Double value) ويفهم(يضبط) على أنك تحاول إسناد قيمة مزدوجة Double إلى متغير عشري Decimal . لتحويل القيم العددية إلى النوع عشري Decimal type استخدم التمثيل التالي Discount = 0.35D

كما سترى فيما بعد الحرف D عند نهاية القيمة الرقمية يدل على أن القيمة ستعامل كقيمة عشرية , يوجد أيضا العديد من الحروف النوعية (type characters) راجع الجدول المرفق ، لقد استخدمت نوع البيانات العشرية هنا لأنها تستخدم بشكل عام في الحسابات التمويلية (financial calculations) إذا أردت أن توفر حسم أفضل كل ما عليك فعله هو تغير القيمة للمتغير Discount فإذا لم تستخدم المتغيرDiscount سيتوجب عليك عمل العديد من التغيرات على كل مكان في كودك . أو بكلمة أخرى إذا كتبت السطر الذي يحسب الكمية المحسومة كما سترى فعليك البحث عن كل الأسطر في كودك التي تحسب الحسومات وتغير الحسم من 0.35 إلى قيمة أخرى :

```
DiscountedAmount = 24500 * (1 - 0.35)
```

فبمجرد تغير قيمة المتغير Discount في مكان واحد ضمن كودك سيتم تحديث كامل البرنامج ((the Entire program is up-to-date.

التصريح(الإعلان)عن المتغيرات Declaring Variables

في معظم لغات البرمجة المتغيرات يجب أن يتم الإعلان عنها في المقدمة ,تاريخيا السبب في عمل هذا هو مساعدة المترجم على إنتاج كود أكثر فعالية،فإذا كان المترجم يعرف جميع المتغيرات وأنواعها في وقت مبكر يستطيع أن ينتج الكود المحكم والأكثر فعالية. أو يعمل على تحسين الكود , فمثلا عندما تخبر المترجم على أن المتغير Discount سيحفظ قيمة رقمية ,فان المترجم سيحجز عدد محدد من البايتات للمتغير Discount كي يستخدمها. ميزة للبيسك BASIC كانت وما تزال من أكثر الانتقادات حدة (intensely criticized) هي أنها لا تجبر المبرمج على التصريح عن جميع المتغيرات, كما سترى ,توجد أسباب لعملية الترجمة من أكثر أهمية من زيادة السرعة والفعالية والتي تجعل من الضروري الإعلان عن المتغيرات,فمثلا عندما يعلم المترجم نوع المتغيرات مقدما in advance يستطيع أن يلتقط العديد من الأخطاء وقت التصميم أو وقت التنفيذ, هذه الأخطاء إن لم تعالج ستطفو على السطح وقت التنفيذ , فعندما تصرح عن متغير من نوع تاريخ Date فان المترجم لن يسمح لك بإسناد قيمة رقمية صحيحة إليه.

عند البرمجة في VB 2008 عليك التصريح عن المتغيرات لأنه النمط الافتراضي ومايكروسوفت توصي بشدة بهذه الممارسة, إذا حاولت أن تستخدم متغير غير مصرح عنه في كودك فان VB 2008 سترمي باستثناء . إنها ستلتقط الخطأ حالما تنتهي من كتابة السطر الذي يستخدم المتغير الغير مصرح عنه وتضع تحته خط متعرج احمر, من الممكن تغير السلوك الافتراضي وتستخدم المتغيرات الغير مصرح عنها وهي الطريقة التي كان معظم الناس في الإصدارات القديمة من VB يعمل بها (سترى كيفية فعل هذا لاحقا في هذا المقطع) ولكن جميع الأمثلة في هذا الكتاب تستخدم المتغيرات المعلن عنها بشكل صريح ، على أية حال عليك وبقوة أن تتشجع للتصريح عن متغيراتك . للإعلان عن متغير ما استخدم العبارة Dim متبوعة باسم المتغير والكلمة المحجوزة As ومن ثم نوعه, كما يلي:

```
Dim meters As Integer
Dim greetings As String
```

المتغير الأول meters سيخزن أعداد صحيحة مثل 3 أو 1002 والمتغير الثاني greetings سيخزن نص , تستطيع أن تصرح عن عدة متغيرات من نفس النوع أو مختلفة في نفس السطر, كما يلي :

```
Dim Qty As Integer, Amount As Decimal, CardNum As String
```

أما للإعلان عن عدد من المتغيرات من نفس النوع ليس عليك تكرار النوع فقط افصل بين جميع المتغيرات التي من نفس النوع لفاصلة commas وضع النوع للمتغير الأخير

```
Dim Length, Width, Height As Integer, Volume, Area As Double
```

هذه العبارة تصرح عن ثلاث متغيرات من النوع عدد صحيح Integer ومتغيران من النوع المزدوج Double . المتغيرات من النوع المزدوج تحفظ القيم الكسرية fractional أو القيم ذات النقطة العائمة(floating-point values) وهو مشابه لنوع البيانات مفرد مع استثناء هو أنها تستطيع أن نمثل القيم الغير صحيحة بدقة أكبرgreater accuracy . تستطيع استخدام كلمات محجوزة أخرى في التصريح عن المتغيرات , مثل Private, Public, and Static وهذه الكلمات المحجوزة تدعى محددات الوصول access modifiers لأنها تحدد أي من مقاطع كودك تستطيع الوصول إليها المتغيرات المخصصة بهذه المحددات وأي من المقاطع لا تستطيع الوصول إليها, في الوقت الحالي ضع في رأسك أن جميع المتغيرات

المصرح عنها بالعبارة Dim تتواجد في التركيبة(module) التي بها تم الإعلان عنها فإذا ما تم الإعلان عن المتغير Countفي الأجراء الفرعي subroutine(معالج الحدث ,مثلا) فانه يتواجد فقط في هذا الإجراء الفرعي ,ولن تستطيع الوصول إلى هذا المتغير من خارج هذا الإجراء الفرعي ، عمليا يمكن إن يكون لديك المتغير Count في عدة إجراءات فرعية وكل متغير هو مخزن محلي ولا يتداخل أحدها مع الأخر.

اصطلاحات تسمية المتغيرات Variable-Naming Conventions

عندما تصرح عن المتغيرات عليك أن تكون مدرك لاصطلاحات(تقاليد) تسمية المتغيرات فأسماء المتغيرات :

- يجب أن تبدأ بحرف letter متبوعا بعدد أخر من الأحرف أو الأرقام digits
- لا تحوي على نقاط ضمنية أو أي من علامات الترقيم punctuation symbols فقط مسموح بميزة الخط المنخفض underscore character
- يجب أن لا تزيد عن 255 حرف characters
- يجب أن تكون مفردة ومميزة ضمن مجالها scope هذا يعني انك لا تستطيع أن يكون لديك متغيران متطابقان في الاسم في نفس الإجراء الفرعي ولكن يمكن ان يكون لديك نفس الاسم في عدة إجراءات فرعية.

اسم المتغير في VB 2008 غير حساسة لحالة الأحرف case-insensitive فجميع هذه المتغيرات myAge, myage, and MYAGE تشير إلى نفس المتغير في بشكل عملي كلما أدخلت أسماء المتغيرات فان المحرر سيحول حالتها بحيث تطابق تصريحاتها.

التمهيد للمتغيرات Variable Initialization

يسمح لك الفيجوال بيسك 2008 بالتمهيد للمتغيرات في نفس السطر الذي صُرّح عنها فيه. العبارة التالية تصرح عن متغير قيمة صحيحة Integer وتمهد له (تسند) القيمة 3,045

```
Dim distance As Integer = 3045
```

وهذه العبارة مكافئة للسطرين التاليين:

```
Dim distance As Integer
distance = 3045
```

ويمكن ان تصرح وتمهد للعديد من المتغيرات التي من نفس النوع أو مختلفة,في نفس السطر

```
Dim quantity As Integer = 1, discount As Single = 0.25
```

النوع (استنتاج)Type Inference

كما ذكرت سابقا واحد من العلامات المميزة للبيسك (trademark features of BASIC) بما فيها الإصدارات السابقة من البيسك هي القدرة على استخدام المتغيرات بدون التصريح عنها ,وهذا التدريب غير مرغوب فيه أبدا, هذه الميزة تعود إلى اللغة, فقط في الحالة الآمنة. يسمح لك الفيجوال بيسك2008 بأن تصرح عن المتغيرات بإسناد قيم لها ,والمترجم سيقوم باستنتاج نوع المتغير من خلال قيمته المسندة و سينشئ متغير ما لنوع محدد خلف المشهد, العبارة التالية تنشئ متغير من Integer :

```
Dim count = 2999
```

لتعرف نوع المتغير, استخدم الطريقة GetType method هذه الطريقة تعود بنوع الكائن الذي يمثل نوع المتغير.اسم النوع يتم الحصول عليه بواسطة الخاصية TOSTRING property العبارة التالية ستطبع النص System.Int32 في نافذة Immediate window:

```
Debug.WriteLine(count.GetType.ToString)
```

المتغير count هو من النوع Integer type فإذا حاولت إسناد قيمة من نوع مختلف إلى هذا المتغير فيما بعد في كودك مثل تاريخ فان المحرر سيضع خط تحت القيمة ويولد التحذير التالي Value of type ‘Date’ cannot be converted to Integer فالمترجم قام باستنتاج نوع القيمة المسندة بشكل أولي إلى المتغير وانشأ متغير موافق للنوع . وهذا هو السبب في أن العبارات المتلاحقة subsequent لا تستطيع تغير نوع المتغير تستطيع أن تعطل النوع inference بإدخال العبارة التالية أعلى الموديل (module)

```
Option Infer Off
```

بشكل اختياري تستطيع آن تفعِّل أو تعطل هذا الخيار من صفحة خصائص المشروع فإذا كان خيار التحويل معطل (off)فان المترجم سيعامل المتغيرات المصرح عنها بدون نوع معين بالاعتماد على خيار التدقيق(option strict) , إذا كان هذا الخيار معطل(off) فان المترجم سينشئ متغير كائن Object variable والذي يستطيع أن يخزن أي قيمة, حتى قيم الأنواع المختلفة في منهج التطبيق . إذا كان الخيار (option strict on) مفعَّل فان المترجم سيرفض الإعلان ,وسيضع خط تحت اسم المتغير مع خط متعرج ويولد التحذير التالي :
Option Strict On requires all variable declarations to have an As clause بشكل عام فاني استخدم الإعلان الصريح والذي يوصى به كأفضل تدريب لإنشاء واستخدام المتغيرات.

أنواع المتغيرات Types of Variables

يتعرف الفيجوال بيسك على خمس أنواع تصنيفات للمتغيرات:

- الرقمي Numeric
- النصي String
- المنطقي Boolean
- التاريخ Date
- النوع العام (كائن) Object

النوعين الرئيسين لتصنيفات المتغيرات هي العددية والنصية فالمتغيرات العددية تخزن الأعداد والمتغيرات النصية تخزن النصوص بينما التغير العام (الكائن) يستطيع تخزين أي نوع من البيانات, لماذا نعاني لتحديد نوع المتغير إذا كان واحد منها يناسب الجميع ؟ بشكل سطحي المتغيرات من النوع كائن يمكن أن تبدو فكرة جيدة ولكن لها مساؤها disadvantages فالمتغيرات للنوع العددي الصحيح Integer variables تم تحسينها للتخزين الأمثل القيم الصحيحة integers ومتغيرات التاريخ تم تحسينها للتخزين الأمثل للتواريخ. قبل أن تستطيع الفيجوال بيسك استخدام المتغير العام يجب أن تحدد نوعه وتنجز التحويل الضروري necessary conversions . إذا تم التصريح عن المتغير بنوع معين فهذا التحويل غير ضروري .

سنبدأ مناقشتنا لأنواع المتغيرات بالمتغيرات العددية,فالأعداد يمكن أن تخزن في عدة تنسيقات بالاعتماد على حجم العدد ودقته .لذا يوجد عدة أنواع للمتغيرات العددية, البيانات من نوع تاريخ ونص هي أغنى بلغة الوظيفة (الفعالية) التي تعرضها.

المتغيرات العددية Numeric Variables

جميع لغات البرمجة تزود بأنواع متنوعة من البيانات العددية متضمنة التالي :

- البيانات الصحيحة Integers (يوجد عدة أنواع للبيانات الصحيحة)
- العشرية Decimals
- المفردة أو الأعداد ذات النقطة (الفاصلة)العائمة بدقة محدودة Single, or floating-point numbers with limited precision
- المزدوجة Double أو الأعداد ذات النقطة(الفاصلة) العائمة بدقة عالية floating-point numbers with extreme precision

العشري والمفرد والمزدوج هي ثلاث أنواع رئيسية من البيانات لتخزين الأعداد ذات الفاصلة العائمة(الأعداد بجزء كسري) فنوع البيانات المزدوج يستطيع أن يمثل هذه الأعداد بدقة accurately أعلى من النوع المفرد ويستخدم بشكل خاص exclusively في الحسابات العملية scientific calculations.
أنواع البيانات الصحيحة تخزن الأعداد الصحيحة whole numbers. نوع بيانات متغيرك يمكن أن تصنع اختلاف في نتيجة الحسابات . الأنواع المناسبة للمتغير يتم تحديدها بواسطة طبيعة القيم التي تمثلها ، واختيار نوع البيانات هو بشكل متناوب (متكرر) تسوية(trade-off) بين الدقة precision والسرعة في التنفيذ execution(نوع البيانات الأقل دقة تكون أسرع في المعالجة less-precise data types are manipulated faster).

المتغيرات العددية الصحيحة integer Variables

يوجد ثلاث أنواع للمتغيرات لتخزين الأعداد الصحيحة ,وهي تختلف في مجال الأعداد التي تستطيع تمثيلها فقط وكما تعلم النوع الذي يأخذ بايتات أكثر يستطيع أن يخزن قيم أكبر . ونوع المتغير الصحيح الذي ستستخدمه يعتمد على المهمة التي تكون قيد الاستخدام ,وسوف تختار النوع الذي يمثل القيم الأكبر فتوقعك المسبق (anticipate)سيظهر في حساباتك,تستطيع الذهاب إلى النوع الطويل Long لتكون أمن ولكن هذا المتغير هو أكبر بأربع مرات من المتغير القصير Short ويأخذ من الكمبيوتر فترة أطول للمعالجة من تلك التي يأخذها المتغير القصير.
كل نوع بيانات رقمية يعرض خاصتي القيمة الصغرى MINVALUE والقيمة العظمى MAXVALUE واللتان تعودان بالقيمة الصغرىminimum والقيمة الكبرى maximum على الترتيب respectively والتي يمكن تمثيلها represented بواسطة نوع بيانات الموافق corresponding فقيم النوع القصير Short (Int16) يمكن تخزينها في المتغير الصحيح(Int32) Integer وفي المتغير الطويل Long (Int64) ولكن العكس(reverse) ليس صحيحا . إذا حاولت تخزين قيمة من النوع الطويل في متغيرInteger سينتج عنه خطأ والمترجم سيضع خط تحت مزعج offending متعرج wiggly ولقد قمت بتضمين عبارات بشرح الخطأ الناتج عن بعض العبارات.

```
Dim shortInt As Int16
Dim Int As Int32
Dim longInt As Int64
 Debug.WriteLine(Int16.MinValue)
 Debug.WriteLine(Int16.MaxValue)
 Debug.WriteLine(Int32.MinValue)
 Debug.WriteLine(Int32.MaxValue)
 Debug.WriteLine(Int64.MinValue)
 Debug.WriteLine(Int64.MaxValue)
 shortInt = Int16.MaxValue + 1
' ERROR, exceeds the maximum value of the Short data type(خطأ,زيادة عن حد القيمة العظمى لنوع البيانات القصيرة)
 Int = Int16.MaxValue + 1
' OK, is within the range of the Integer data type( صحيح ضمن مجال نوع البيانات الصحيحة (الإنتغر32 )
 Int = Int32.MaxValue + 1
' ERROR, exceeds the maximum value of the Integer data type(خاطئة,زيادة عن حد القيمة العظمى لنوع البيانات integer)
 Int = Int32.MinValue - 1
' ERROR, exceeds the minimum value of the Integer data type(خاطئة,زيادة عن حد القيمة الصغرى لنوع البيانات integer )
 longInt = Int32.MaxValue + 1
 ' OK, is within the range of the Long data type( صحيح ضمن مجال نوع البيانات الطويل (64 )
 longInt = Int64.MaxValue + 1
' ERROR, exceeds the range of all Integer data types (خاطئة,زيادة عن حد مجال جميع البيانات الصحيحة )
```

عبارات WriteLine الستة ستطبع القيم العظمى والصغرى التي تستطيع تمثيلها بأنواع بيانات الأعداد الصحيحة المتنوعة , العبارات التي تليها مشروحة بواسطة التعليقات ذات الخط الأخضر والتي يقابلها معنى التعليق في اللغة العربية بالخط الأحمر, فإذا ما حركت المؤشر فوق العبارات الخاطئة مثلا سترى وصف لرسالة الخطأ كما يلي بالنسبة للعبارة الأولى : Constant expression not representable in type Short(لا يمكن تمثيل الثابت في النوع قصير (short)), حاول قراءة وفهم باقي الكود مع التعليقات.

الجدول التالي يبين أنواع البيانات العددية للفيجوال بيسك

نوع البيانات data type	تمثيله في الذاكرة	المجال(السعة التخزينية)
Byte (Byte)(بايت واحد)	1 byte	أعداد صحيحة في المجال من 0 إلى 255
Signed Byte (sbyte)(بايت واحد باشارة)	1 byte	أعداد صحيحة في المجال من -128 إلى 127
Short (Int16)(قصير16بت)	2 byte	قيم صحيحة في المجال من -32،768 إلى 32،767
Integer (Int32)(صحيح 32 بت)	4 byte	قيم صحيحة في المجال من −2,147,483,648 إلى 2,147,483,647
Long (Int64)(طويل64 بت)	8 byte	قيم صحيحة في المجال من −9,223,372,036,854,755,808 إلى 9,223,372,036,854,755,807
Unsigned Short (UShort)(قصير بدون إشارة 16 بت)	2 byte	قيم صحيحة موجبة في المجال من 0 إلى 65,535
Unsigned Integer (UInteger)(صحيح بدون إشارة 32 بت)	4 bytes	قيم صحيحة موجبة في المجال من 0 إلى 4,294,967,295
Unsigned Long (ULong)(طويل بدون إشارة 64 بت)	8 bytes	قيم صحيحة موجبة في المجال من 0 إلى 18,446,744,073,709,551,615
Single Precision (Single)(مفرد(الكسرية بدقة مفردة))	4 bytes	أعداد ذات فاصلة عائمة بدقة مفردة وتمثل الأعداد السالبة في المجال من −3.402823E38 إلى −1.401298E-45 والأعداد الموجبة في المجال من 1.401298E-45 إلى 3.402823E38 والقيمة 0 لايمكن تمثيلها بدقة(هي عدد صغير ،صغير جدا ولكن ليست صفر تماما)
Double Precision (Double)(مزدوج(الكسرية بدقة مزدوجة)	8 bytes	أعداد ذات فاصلة عائمة(الكسرية) بدقة مزدوجة وتمثل الأعداد السالبة في المجال من −1.79769313486232E308 إلى −4.94065645841247E-324 والأعداد الموجبة في المجال من 4.94065645841247E-324 إلى 1.79769313486232E308
Decimal (Decimal)(الأعداد العشرية(الصحيحة و الكسرية)	16 bytes	الأعداد الصحيحة و الكسرية(ذات الفاصلة العائمة مدرجة بواسطة عامل تحليل في المجال من 0 إلى 28)

ملاحظة: 1بايت(byte) = 8 بت(bit) يتم تخزين الحرف في 8 بت (نبضات) فالبايت وحدة قياس المساحة وتساوي حرف واحد أما البت فهو وحدة مساحات التخزين و هو أصغر وحدة قياس للمعلومات في الحاسب

يقسم جدول الأسكي إلى ثلاث مناطق :

1- المنطقة من 1 إلى 31 تحوي على رموز لا يمكن طباعتها بل تحوي بعض الأشياء مثل علامة بداية السطر

2- المنطقة من 32 إلى 127 تحوي الأبجدية الانكليزية والرموز الشائعة

3- الأعلى من 127 تحوي على حروف غير الانكليزية في الويندوز العربي تكون هذه الحروف عربية.

وكذلك الأمر بالنسبة إلى اليونيكود ولكن حروف اليونيكود (Unicode characters) يتم تخزينها في 2 بايت

الأعداد الكسرية ذات الدقة المفردة والمزدوجة Single- and Double-Precision Numbers

إن الأسماء مفرد Single ومزدوج Double أتت من دقة مفردة single-precision ودقة مزدوجة double-precision للأعداد . فالأعداد ذات الدقة المزدوجة يتم تخزينها بشكل داخلي بدقة أعلى من الأعداد ذات الدقة المفردة . في الحسابات العلمية تحتاج إلى كامل الدقة التي تستطيع الحصول عليها وفي هذه الحالات عليك استخدام البيانات من النوع المزدوج.

فنتيجة العملية 3 / 1 هو . . . 0.333333 (عدد غير منته من الرقم 3) تستطيع مليء 256 ميغا بايتMB256 من الرام RAM بالأرقام 3 والنتيجة ستبقى مبتورة (غير كاملة truncated) إليك مثال يوضح تأثيرات القطع(البترtruncation) : في معالج حدث الضغط على الزر صرح عن متغيرين كما تعلمت سابقا وكما يلي :

```
Dim a As Single, b As Double
```

ومن ثم ادخل العبارات التالية :

```
a = 1 / 3
Debug.WriteLine(a)
```

نفذ التطبيق وستحصل على النتيجة التالية في نافذة الفورية immediat :

.3333333

يوجد سبع أرقام على يمين الفاصلة العشرية decimal point , أوقف التطبيق بالضغط على Ctrl+ Break وقم بإلحاق الأسطر التالية في نهاية مقطع الكود السابق:

```
a = a * 100000
Debug.WriteLine(a)
```

هذه المرة ستحصل على النتيجة التاليةفي نافذة immediat :

33333.34

هذه النتيجة غير دقيقة كما يمكن أن تكون قد توقعت بشكل أولي وحتى لم يتم تدويرها بشكل مناسب rounded فإذا ما قسمت a على 100,000 ستكون النتيجة: 0.3333334

هذا العدد مختلف عن العدد الذي بدأنا به وهو (0.3333333). فالقيمة الأولية تم تدورها عندما ضربت بـ multiplied it by 100,000 وتم تخزينها في متغير مفرد Single variable وهذه نقطة مهمة في الحسابات العددية, وتسمى error propagation(تكاثر الخطأ). في السلاسل الطويلة للحسابات العددية فان الأخطاء ستتكاثر errors propagate ، حتى ولو تستطيع أن تجيز(tolerate) الخطأ المقدم بواسطة نوع البيانات المفرد Single في عملية وحيدة فان الأخطاء التراكمية(cumulative errors) يمكن أن تصبح ذات معنى هام(مؤثرة significant) . دعنا ننجز نفس العملية ولكن بأعداد ذات دقة مزدوجة (double-precision numbers) في هذه المرة استخدم المتغير b. أضف السطور التالية إلى معالج حدث ضغط الزر:

```
b = 1 / 3
Debug.WriteLine(b)
b = b * 100000
Debug.WriteLine(b)
```

هذه المرة سيتم عرض الأعداد التالية في النافذة immediat : 0.333333333333333, 33333.3333333333

النتائج المنتجة بواسطة المتغيرات ذات ا لدقة المزدوجة أكثر دقة.
لماذا مثل هذه الأخطاء تنتج في حساباتنا؟
إن السبب في ذلك هو أن الكمبيوتر يخزن الأعداد بشكل داخلي بواسطة رقمان وهما:الصفر0 والواحد 1 . وهذا مناسب(مريح convenient) جدا للكمبيوتر, لأن الإلكترونيات تفهم حالتان فقط (on and off), في الواقع As a matter of fact, جميع العبارات يتم ترجمتها إلى بيتات bits(أصفار و وحدات zeros and ones) قبل أن يفهمها الكمبيوتر وينفذها.
نظام العد الثنائي (binary numbering system) المستخدم بواسطة الكمبيوترات لا يختلف كثيرا عن النظام العشري decimal system الذي نستخدمه نحن البشر, فالكمبيوتر فقط يستخدم القليل من الأرقام,بينما نحن البشر نستخدم عشرة أرقام مختلفة لتمثيل الأعداد,الصحيحة أو الكسرية (whole or fractional) لأن لدينا عشرة أصابع(بشكل مساير,للكمبيوتر فقط إصبعان يستطيع العد بهما) وكما في نظام العد العشري decimal numbering system والذي به بعض الأعداد لا يمكن تمثيلها بدقة, يوجد أيضا أعداد لا يمكن تمثيلها بدقة في النظام الثنائي binary system.
دعني أقدم لك مثال أكثر توضيحا illuminating example . قم بإنشاء متغير a ذو دقة مفردة single-precision variable ومتغير b ذو دقة مزدوجة double-precision variable واسند نفس القيمة إليهم :

```
Dim a As Single, b As Double
a = 0.03007
b = 0.03007
```

ومن ثم أطبع الاختلاف بينهما :

```
Debug.WriteLine(a - b)
```

إذا نفذت هذه الأسطر فان النتيجة لن تكون صفرا وإنما ستكون 6.03199004634014E-10- وهذا عدد صغير جدا ويمكن كتابته ايضا كما يلي 0.00000000060319900463401 لان الأنواع الرقمية المختلفة تخزن بشكل مختلف في الذاكرة ولا تتطابق تماما,ما يعني هذا لك أن جميع المتغيرات في حساب ما يجب أن تكون من نفس النوع.
أخيرا Eventually الكمبيوترات ستفهم مجموعة الرموز الرياضية mathematical notation ولن تحول جميع التعابير العددية إلى قيم , كما هي تفعل اليوم . إذا ضربت التعبير 1/3 بالعدد 3 فان النتيجة لن تكون واحد(1) .
الكمبيوترات مهما يكن عليها تحويل التعبير 1/3 إلى قيمة قبل أن تستطيع ضربها بـ3 , لان 1/3 لا يمكن تمثيلها بدقة ,فالنتيجة لـ3 × (1/3) لن تكون واحد تماما, .
إذا تم التصريح عن المتغيرين a و b كمفرد Single أو مزدوج Double فان العبارات التالية سوف تطبع واحد,كما يلي: (النتيجة واحد في حال البيانات من نفس النوع)

```
a = 3
b = 1 / a
Debug.WriteLine(a * b)
```

إذا تم التصريح عن المتغيران السابقان كنوع عشري as Decimal فان النتيجة ستكون رقم قريب إلى الواحد ولكن ليس واحد تماما وإنما (0.9999999999999999999999999999) يوجد حوالي 28 رقم بعد الفاصلة العشرية .

نوع البيانات العشرية The Decimal Data Type

المتغيرات من النوع العشري تخزن داخليا كـــ integers (قيم صحيحة) في 16بايت(16 bytes) وتدرج (تقسم(بأس)أو بقوة الأساس 10(10 is scaled by a power of) , أس التدريج(التقسيم) scaling power يحدد عدد الأرقام العشرية التي على يمين الفاصلة العائمة ,وهو قيمة صحيحة من 0 إلى 28 عندما تكون قوة التدريج 0 صفر فان القيمة تضرب بواحد(1) أو 10^0 (عشرة قوة(أس)صفر) ويتم تمثيلها بدون رقم عشري . بينما عندما يكون الأس(قوة التدريج(قوة العدد 10) هو 28 فان القيمة تقسم على 10^28 والتي هي واحد متبوع بـ 28 صفر(وهي قيمة ضخمة) ويتم تمثيلها بــــ 28 رقم عشري .
القيمة الأكبر التي يمكن أن يتم تمثيلها بالقيمة العشرية هي الصحيحة (integer 79,228,162,514,264,337,593,543,950,335) والقيمة الأصغر التي يمكن أن تمثلها بالمتغير العشري هي نفس القيمة السابقة مع إشارة السالب , وهذه القيم تستخدم عامل (أس) التدريج صفر(0) أي (10^0), عندما يكون عامل الموازنة هو 28 فان القيمة الأعلى التي يمكن لك أن تمثلها في المتغير العشري صغيرة تماما عمليا هي 7.9228162514264337593543950335(والقيمة الأصغر نفسها مع إشارة سالب). وهذه قيمة عددية صغيرة جدا (وليست تماما 8) ولكنها مثلت بزيادة مفرطة في الدقة extreme accuracy.
العدد صفر لايمكن تمثيله بدقة بواسطة متغير عشري مدرج بالعامل factor)(الأس) 28 , القيمة الموجبة الأصغر التي تستطيع تمثيلها بنفس أس التدريج هي (01. . .0.00)(يوجد 27 صفر بين الفاصلة العشرية و الرقم واحد 1) ـ قيمة صغيرة للغاية ولكن تبقى تماما ليست صفرا .كلما زادت الدقة التي تريد أن تحققها بواسطة المتغير العشري كلما صغر مجال القيم الممكنة المتوفرة تحت تصرفك(you have at your disposal), تماما مثل كل شيء أخر موجود في الحياة.
عند استخدام الأعداد العشرية ,فان المترجم يحفظ مسار الأرقام العشرية (الأرقام التي تتبع النقطة العشرية) ويعامل جميع القيم كقيم صحيحة integers فالقيمة 235.85 يتم تمثيلها كقيمة صحيحة 23585 ولكن المترجم يعرف أنه يتوجب عليه تخفيض القيمة(scale down) بالنسبة 100(تقسيم القيمة على 10^2)عندما ينتهي من استخدامها .
التقسيم (التخفيض بالنسبة 100)والتي هي(102) مطابق لرفع الفاصلة العشرية موضعين(رقمين) ,فأولا يضرب المترجم هذه القيمة بمائة(100) لجعلها صحيحة ومن ثم يقسمها على 100 لإعادة القيمة إلى قيمتها الأصلية, دعنا نقول انك تريد أن تضرب القيم التالية : 328.558 * 12.4051
عليك أولا أن تحولهم إلى قيم صحيحة integers وعليك أن تتذكر أن الرقم الأول لديه ثلاث أرقام عشرية,والثاني لديه أربع أرقام عشرية . ونتيجة الضرب سيكون لديها سبع أرقام عشرية ,لذلك تستطيع أن تضرب القيم التالية: 328558 * 124051 ومن ثم تعامل الأرقام السبع الأخيرة للنتيجة كأرقام عشرية, استخدم حاسبة ويندوز (في الوضع العلمي Scientific view) لحساب الناتج السابق والنتيجة هي 40,757,948,458 القيمة الحقيقية بعد الأخذ بعين الاعتبار الأرقام العشرية هي 4,075.7948458 هذه هي الطريقة التي يعالج فيها المترجم البيانات من النوع العشري Decimal data type ., أدخل السطور التالية في معالج حدث الضغط على الزر ونفذ البرنامج:

```
Dim a As Decimal = 328.558D
Dim b As Decimal = 12.4051D
Dim c As Decimal
c = a * b
Debug.WriteLine(c.ToString)
```

الحرف D عند نهاية العبارتين يحدد على أن الأعداد سوف تحول converted إلى قيم عشرية, بشكل افتراضي كل قيمة بجزء كسري (fractional part) تعامل كقيمة مزدوجة Double . إسناد قيمة من النوع المزدوج إلى متغير عشري سوف ينتج عنه خطأ إذا كان خيار التدقيق فعال(يعمل Strict option is on) لذا عليك أن تحدد بشكل صريح أن القيمتان سيتم تحويلهما إلى النوع العشري Decimal type الحرف D عند نهاية القيمة تدعى (حرف النوع type character.).
إذا نفذت نفس الحساب السابق باستخدام متغير من النوع المفرد فان النتيجة ستكون مبتورة truncated(ومدورة) إلى ثلاث أرقام عشرية 4,075.795 لاحظ أن نوع البيانات العشرية لا تقدم أي خطأ نتيجة التدوير .لديها المقدرة على تمثيل النتيجة بنفس العدد من الأرقام العشرية, وهذه هي الفائدة الحقيقية للنوع العشري, والذي يجعل منها النوع المثالي ideal لتطبيقات التمويل financial applications . من أجل الحسابات العلمية يتوجب عليك استخدام النوع المزدوج Doubles .بينما الأعداد العشري هي الاختيار الأفضل للحسابات التي تتطلب دقة خاصة (مثل ثمانية أرقام عشرية)

جدول الحروف النوعية(Type Characters)لتحويل الأنواع المختلفة للبيانات

حرف النوع	الوصف	مثال
C	تحول القيمة إلى النوع الحرفي char	Dim ch As String = "A"c

Dim price As Decimal = 12.99D	تحول القيمة إلى النوع العشري Decimal type	D or @
Dim pi As Double = 3.14 R	تحول القيمة إلى النوع المزدوج Double type	R or #
Dim count As Integer = 99I	تحول القيمة إلى النوع الصحيح Integer type	I or %
Dim distance As Long = 1999L	تحول القيمة إلى النوع الطويل Long type	L or &
Dim age As Short = 1S	تحول القيمة إلى النوع القصير Short type	S
Dim velocity As Single = 74.99F	تحول القيمة إلى النوع المفرد Single type	F or !

اللانهاية و الشذوذات الأخرى Infinity and Other Oddities

يمكن لمنصة Frameworkالعمل أن تمثل اثنان من القيم الخاصة جدا والتي يمكن أن لا تكون قيم عددية بحد ذاتها ولكن ناشئة عن حسابات عددية: NAN(ليس رقم) واللانهاية (Infinity), إذا أنتجت حساباتك NAN أو اللانهاية, عليك تأكيد بياناتك (التحقق من بياناتك confirm) وأعمل على إعادة الحسابات, أو استسلم . من أجل جميع أهداف العملية, فان لا NAN ولا حتى اللانهاية يمكن أن تستخدم في الحسابات العملية اليومية ,

ملاحظة

ليس عدد(ل.ع Not a Number (NAN)

ل.ع (NAN) ليست جديدة فحزم مايكروسوفت اكسل قد استخدمتها لسنوات و ماتزال, تدل القيمة ل.ع(NAN) على أن نتيجة عملية ما لايمكن تعريفها إنها ليست عدد نظامي, ليس صفر وليست لانهاية .

ل.ع هي أكثر من كونها مبدأ رياضي يحل محل قيمة تستطيع استخدامها في حساباتك. فالوظيفة Log() function تحسب لوغاريتم القيم الموجبة, وبشكل محدد لا تستطيع حساب لوغاريتم الأعداد السالبة ,فإذا مررت معامل نسبي سالب إلى الوظيفة فان الوظيفة المذكورة ستعود بقيمة ل.ع لتدل على أن الحسابات قد أنتجت قيمة غير صحيحة ,يمكن أن تجد إنها مزعجة annoying فوظيفة عددية تعود بقيمة غير عددية,ولكن هذا أفضل من رمي استثناء(throwing an exception),حتى ولو لم تضبط هذه الحالة مباشرة, فان حساباتك ستسمر وسينتج عنها جميعها القيم ل.ع.

بعض الحسابات تعطي نتائج غير معروفة مثل اللانهاية ,رياضيا: نتيجة تقسيم أي عدد على الصفر هو لانهاية ,لسوء الحظ الكمبيوترات لا تستطيع تمثيل اللانهاية لذلك فإنها تنتج خطأ عندما تطلب التقسيم على صفر , الفيجوال بيسك 2008 سوف يخبر(يقدم تقريرا) عن قيمة خاصة والتي هي ليست رقم:☹(القيمة لانهاية) .إذا استدعيت الطريقة TOSTRING method لهذه القيمة ستعود بالنص:Infinity . دعنا نولد قيمة لانهاية, ابدأ التصريح عن متغير مزدوج Double variable كما في العبارات التالية :

```
Dim dblVar As Double = 999
Dim infVar As Double
infVar = dblVar / 0
MsgBox(infVar)
```

النص Infinity سيظهر في صندوق رسالة وهذا النص فقط وصف, يخبرك أن النتيجة ليست صحيحة (إنها قيمة ضخمة جدا تزيد عن مجال القيم الرقمية والتي يمكن تمثيلها في أي نوع من البيانات) ولكن لن يتم استخدامها في الحسابات الأخرى ,مهما يكن تستطيع استخدام القيمة لانهاية في عمليات حسابية . فعمليات خاصة باللانهاية تكون مفهومة make sense وأخرى لا تكون .

توجد حسابات أخرى ينتج عنها yield ليس عدد non-number فتقسيم عدد كبير على أخر صغير جدا ستزيد النتيجة عن قيمة أي مجال لأنواع البيانات التي يمكن أن تمثل القيمة والنتيجة ستكون لانهاية, صرح عن ثلاث متغيرات في معالج حدث النقر على الزر كما يلي:

```
Dim largeVar As Double = 1.0E+299
Dim smallVar As Double = 1.0E-299
Dim result As Double
result = largeVar / smallVar
MsgBox(result)
```

الرمز 1E299 يعني 10 مرفوعة للأس 299 والذي هو رقم كبير جدا وبنفس الطريقة الرمز 1E-299 يعني 10 مرفوعة للأس 299- والذي يكافئ تقسيم 10 على رقم بضخامة 1E299.

ستكون النتيجة لانهاية, إذا عكست العملية فان النتيجة ستكون صفر وهي ليست صفر تماما ولكن النوع المزدوج لا يستطيع تمثيل القيم العددية القريبة جدا من الصفر بدقة .
تستطيع أيضا توليد القيمة لانهاية بضرب عدد كبير جدا (أو عدد صغير جدا)عدة مرات بنفسه . إن التقسيم 0 / 0 مثلا ليس بقيمة رقمية . للتقسيم على الصفر جرب العبارة التالية:

```
Dim var1, var2 As Double
Dim result As Double
var1 = 0
var2 = 0
result = var1 / var2
MsgBox(result)
```

إذا نفذت العبارة السابقة ستكون النتيجة (ليس برقم.NAN)
العبارة التالية أيضا ستعطي لانهاية:

```
Dim result As Double
    result = result + result
    result = 10 / result
    result = result + 1.0E+299
    MsgBox(result)
```

اختبار التطبيقات لاستكشاف القيم لانهاية و ل.ع Testing for Infinity and NAN

لاستكشاف فيما إذا نتيجة عملية ما هي NaN أو لانهاية استخدم الطريقة IsNaN والطريقة IsInfinity لنوع البيانات المفردة والمزدوجة. البيانات الصحيحة Integer لا تدعم هذه الطرق حتى ولو كان من المحتمل أن تولد لانهاية و ل.ع . فإذا ما عادت الطريقة IsInfinity ب True تستطيع أيضا اختبار إشارة قيمة اللانهاية بواسطة الطرق IsNegativeInfinity and IsPositiveInfinity . في معظم الحالات ستعرض تحذير وتنهي الحسابات, فالعبارات في القائمة اللاحقة تفعل ذلك, ضع هذه العبارات في معالج حدث النقر على الزر وشغل التطبيق.

```
Dim var1, var2 As Double
    Dim result As Double
    var1 = 0
    var2 = 0
    result = var1 / var2
    If Double.IsInfinity(result) Then
      If Double.IsPositiveInfinity(result) Then
        MsgBox("Encountered a very large number. Can't continue")
      Else
        MsgBox("Encountered a very small number. Can't continue")
```

```
        End If
    Else
        If Double.IsNaN(result) Then
            MsgBox("Unexpected error in calculations")
        Else
            MsgBox("The result is : " & result.ToString)
        End If
    End If
```

المتغيرات بايت Byte Variables

لا يوجد أي من لأنواع السابقة يتم تخزينه في بايت مفرد . في بعض الحالات يتم تخزين البيانات كبايتات ويتوجب عليك التمكن من الوصول إلى البايتات المفردة(الحرة), نوع البيانات بايت تحفظ القيم الصحيحة في المجال من 0 إلى 255 . وتستخدم البايتات بشكل متكرر للتمكن من الوصول إلى الملفات الثنائية مثل الصور image وملفات الصوت sound files ., لاحظ انك لن تستخدم البايتات للتمكن من الوصول إلى الحروف المفردة, فحروف اليونيكود (Unicode characters) يتم تخزينها في 2 بايت. للتصريح عن متغير كبايت استخدم العبارة التالية:

```
Dim n As Byte
```

المتغير n يمكن أن يستخدم في الحسابات العددية أيضا ,ولكن كن حذر ولا تسند النتيجة إلى متغير أخر من النوع بايت فيمكن لقيمتها ان تتجاوز المجال للنوع البايت., اذا مهد لمتغيرين من النوع بايت كما يلي :

```
Dim A As Byte, B As Byte
A = 233
B = 50
```

فان العبارة التالية سوف ينتج عنها استثناء تجاوز الحد overflow exception

```
Debug.WriteLine(A + B)
```

والعبارة التالية ستظهر في نافذة immediat window

A first chance exception of type 'System.OverflowException' occurred in WindowsApplication1.exe

ونفس الامر سيحدث اذا حاولت اسناد قيمة للمتغير بايت بالعبارة التالية: B = A + B فالنتيجة هي 283 لا يمكن تخزينها في بايت واحد فيجاوب بيسك يولد الجواب الصحيح ولكنه لا يستطيع أن يخزنها في المتغير بايت.

ملاحظة

العمليات المنطقية بالمتغيرات بايت Boolean Operations with Bytes

العمليات التي لن تسبب تجاوز الحد overflows في المعاملات المنطقية مثل AND, OR, NOT, and XOR والتي تستخدم بشكل متكرر مع المتغيرات من النوع بايت. وهذه ليست معاملات منطقية تعود بصح أو خطأ (True or False) إنها تعمل على دمج البتات المطابقة في كلا المعاملين(المعاملين الذين على طرفي أحد هذه المعاملات AND, OR, NOT, and XOR) وتعود ببايت أخر.فإذا جمعت العددين 199و200 بالمعامل AND فان النتيجة هي 192 فالقيمتان في التنسيق الثنائي هما 11000111و11001000 على الترتيب فإذا أنجزت طريقة مماثلة (مطابقة البتات bitwise أي 0 و0=0 ,1و0=0 ,1و1=1) للمعامل AND على هاتين القيمتين فان النتيجة ستكون 11000000والتي هي القيمة العشرية 192 .

بالإضافة إلى نوع البيانات بايتByte data type تزود VB 2008 بنوع البيانات Signed Byte data type(بايت بإشارة) SByte والذي يستطيع ان يمثل القيم ذات الإشارة في المجال من−128 إلى 127. البايتات تبدأ ببت مفرد 1 bit والذي يمثل القيمة السالبة ومجال القيم الموجبة أصبح اقل بواحد من مجال القيم السالبة لأن القيمة 0 تعتبر قيمة موجبة (حيث البت الأول للقيم الموجبة هو0)

المتغيرات المنطقية Boolean Variables

البيانات من نوع المنطقي تخزن القيم (صح/خطأ True/False) فالمتغيرات المنطقية في جوهرهاin essence هي integers (صحيحة)والتي تأخذ القيمة −1(لصح True) والقيمة (0 لخطأ False) بشكل عملي أي قيمة غير الصفر تعتبر قيمة صحيحة , يتم التصريح عن المتغيرات من النوع المنطقي كما يلي:

```
Dim failure As Boolean
```

يتم التمهيد لها إلى False وتستخدم المتغيرات المنطقية في اختبار الشروط أو الحالات,مثل التالي :

```
Dim failure As Boolean = False

' other statements ...
If failure Then MsgBox("Couldn't complete the operation")
```

يمكن أيضا دمجهم بالمعاملات المنطقية And, Or, Not, and Xor فالمعامل Not يمفصل القيمة toggles (تعمل كمفصل صح/خطأ)للمتغير المنطقي . فالعبارة التالية مفصل(toggle)running = Not running فإذا كان المتغير running هو صح True تعيد إعداده إلى False والعكس بالعكس . هذه العبارة هي الطريق الأقصر في التشفير:

```
Dim running As Boolean
If running = True Then
running = False
Else
running = True
End If
```

المعاملات المنطقية يتم اجراؤها على المتغيرات المنطقية وتعود بمعامل منطقي أخر كنتيجة لهم, العبارات التالية ستعرض رسالة اذا واحد(أو كلا) المتغيران ReadOnly and Hidden صحيحان(True) (على مايبدو(حسب الافتراض)presumably هذه المتغيرات تمثل المواصفات الموافقة لملف ما)

```
If ReadOnly Or Hidden Then
    MsgBox("Couldn't open the file")
  Else
{ statements to open and process file}
End If
```

ان الشرط في عبارة if يدمج قيمتين من النوع المنطقي بواسطة المعامل OR فاذا كان واحد أو كلاهما صحيح فان التعبير الكلي هو صحيح (التعبيرالموضوع ضمن الأقواس(parenthesized) هذه القيمة تنفى negated بواسطة المعامل Not operator والشرط If clause يتم تنفيذه فقط اذا كانت نتيجة النفي negation هي صحيحة True , اذا كان المتغير ReadOnly صح(True)والمتغيرHidden (خطأis False) ستم تقيم التعبيركمايلي (اذا ليس(صح أو خطأ) If Not (True Or False) والقوس(صح أو خطأ)=صح وبالتالي العبارة السابقة تصبح كما يلي:

If Not True والتي (ليس صحيح)=خطأ(False)

المتغيرات النصية String Variables

تخزن المتغيرات النصية فقط النصوص ,والمتغيرات النصية يتم الإعلان عنها كما يلي: Dim someText As String تستطيع إسناد أي نص للمتغيرsomeText تستطيع تخزين تقريبا 2 غيغابايت من النصوص في المتغير النصي (والذي يقابل 2 بليون حرف) و الاسنادات التالية كلها ممكنة:

```
Dim aString As String
    aString = "Now is the time for all good men to come " & " to the aid of their country"
aString = ""
aString = "There are approximately 25,000 words in this chapter"
aString = "25,000"
```

الإسناد الثاني ينشئ نص فارغ,والأخير ينشئ نص والذي يحدث لان يحتوي على أرقام عددية والتي هي أيضا حروف ., الاختلاف بين هذين المتغيرين هو أنها كلا منها يخزن قيمة مختلفة عن الأخرى

```
Dim aNumber As Integer = 25000
Dim aString As String = "25,000"
```

المتغير aString يخزن الحروف 2,5,الفاصلة,0,0 و0 بينما المتغير aNumber يخزن قيمة عدد واحد(مفرد), ولكن يمكن أن تستخدم المتغير aString في الحسابات العددية والمتغير aNumber في العمليات النصية ,حيث أن الفيجوال بيسك سينجز التحويلات الضرورية necessary conversions طالما ان الخيار Strict option معطل (off) البيانات النصية وطرق معالجة النصوص تم مناقشتها ف الفصل الثالث عشر.

المتغيرات الحرفية Character Variables

المتغيرات الحرفية تخزن حرف يونيكود مفرد في 2 بايت(2 bytes),في الواقع الحروف هي أعداد صحيحة قصيرة بدون إشارة((Unsigned Short integers (UInt16),تستطيع استخدام الوظيفة CChar() function لتحويل الأعداد الصحيحة إلى حروف واستخدام الوظيفة CInt() functionلتحويل الحروف إلى القيمة العددية الصحيحة الموافقة,للتصريح عن متغير حرفي استخدم الكلمة المحجوزة Char .Dim char1, char2 As Char

تستطيع التمهيد للمتغيرات الحرفية بإسناد إما حرف أو نص لها,في الحالة الحرفية يتم إسناد فقط الحرف الأول من النص إلى المتغير فالعبارة التالية ستطبع الحروف a و A الى المخرجات :

```
Dim char1 As Char = "a", char2 As Char = "ABC"
Debug.WriteLine(char1)
Debug.WriteLine(char2)
```

وهذه العبارات ستعمل فقط إذا كان الخيار Strict option معطل ,فإذا كان غير معطل فان القيم المسندة إلى المتغيرات char1 and char2ستعلم بعلامة خطأ ولإصلاح الخطأ الذي يمنع من ترجمة الكود غير عبارة التصريح Dim كما يلي : Dim char1 As Char = "a"c, char2 As Char = "A"c

عندما يكون الخيار مفعل لاتستطيع اسناد نص الى متغير حرفي وتتوقع ان يتم استخدام فقط الحرف الأول من النص ,القيم العددية الصحيحة التي توافق الحروف الأنكليزية هي أكواد الانسي ANSI (والتي تعني المعهد القياسي الدولي الأمريكي (Americam Ntional Standard Institute)) للحروف المكافئة , العبارة التالية ستطبع القيمة 65 :

```
Debug.WriteLine(Convert.ToInt32("A"c))
```

اذا حولت الحرف الإغريقي α إلى عدد صحيح فان قيمته هي 945 قيمة ليونيكود Unicode value للحرف الشهير π هي 960 , المتغيرات الحرفية تستخدم في عمليات الربط بالنصوص. ونادرا ما تحفظ البيانات الحقيقية كحروف ,ولكن يمكن أن يكون عليك معالجة الحروف المستقلة في النصوص ,من وقت لأخر . نوع البيانات Char data type تعرض عدد من الطرق المهمة لمعالجة الحروف ولقد تم شرحها في التفصيل في الفصل الثالث عشر., دعنا نقول أن المتغير النصي password يحفظ كلمة سر مستخدم جديد وأنت تطلب كلمة السر تلك على أن تحوي على الأقل رمز خاص واحد ,مقطع الكود التالي يمسح كلمة المرور ويرفضها إذا كانت تحوي على حرف هجائي و أرقام فقط

```
Dim password As String, ch As Char
      Dim i As Integer
      Dim valid As Boolean = False
 While Not valid
            password = InputBox("Please enter your password")
            For i = 0 To password.Length - 1
               ch = password.Chars(i)
               If Not Char.IsLetterOrDigit(ch) Then
                  valid = True
                  Exit For
               End If
            Next
            If valid Then
               MsgBox("You new password will be activated immediately!")
            Else
               MsgBox("Your password must contain at least one special symbol!")
            End If
End While
```

إذا كنت غير مطلع على تراكيب العبارات :If. . .Then, For. . .Next, or While. . .End While تستطيع قراءة الشرح في الفصل التالي ,الكود يطلب من المستخدم إدخال كلمة المرور من خلال صندوق الإدخال input box ,المتغيرvalid هو متغير منطقي وتم التمهيد له بالقيمة False(وليس عليك في الواقع التمهيد للمتغيرات المنطقية إلى خطأ False لأنها القيمة الافتراضية التمهيدية(الأولية) لها ولكن فقط لجعل الكود أكثر قابلية للقراءة فاني أسندت له القيمة False,) وتم وضعه لصح True من ضمن كتلة الحلقة loop فقط إذا كانت كلمة المرور تحوي على قيمة حرفية character والتي ليست حرف(أبجدي) هجائي letter أو رقم ..نحن قمنا بإعدادها بشكل أولي إلى خطأ, لذلك فان الحلقة While. . .End While ستنفذ على الأقل مرة واحدة ,وسوف تستمر في الطلب من المستخدم حتى يدخل كلمة مرور صحيحة ,. الحلقة For. . .Next تمسح المتغير النصي password, حرف حرف في كل مرة ,لدى عملية الدوران التي تقوم بها هذه الحلقة فان الحرف التالي سيتم نسخه إلى المتغير الحرفي ch ,الخاصية Chars property لنوع البيانات النصية هي مصفوفة تحفظ الحروف المستقلة في النص (وهي مثال أخر لفعالية البناء في نوع البيانات)

ومن ثم يختبر البرنامج الحرف الحالي والطريقة IsLetterOrDigit method لنوع البيانات الحرفية تعود بصح إذا كانت القيمة الحرفية إما حرف أبجدي أو رقم فإذا كانت القيمة الحرفية الحالية رمز فان البرنامج يعمل على اعداد المتغير valid إلى صح True,لذلك فان الحلقة الخارجية لن يتم تنفيذها مرة أخرى,وسوف يخرج البرنامج من الحلقة FOR .أخيرا سيطبع الرسالة المناسبة وإما أن يطلب من المستخدم كلمة مرور أخرى أو يغادر(يخرج (quits الفئة Char وطرقها تم شرحها بتفاصيل أكثر في الفصل الثالث عشر,

متغيرات التاريخ Date Variables

قيم للتاريخ والوقت تخزن بشكل داخلي في تنسيق خاص,ولكن لا تحتاج إلى معرفة هذا التنسيق بالضبط ,وهي أعداد بدقة المزدوج ,الجزء الصحيح يمثل التاريخ والجزء الكسري يمثل الوقت .ويصرح عن متغير ما كتاريخ (والذي يمكن أن يخزن كلا من قيمة التاريخ والوقت) كما يلي: Dim expiration As Date والاسنادات التالية كلها صحيحة:

```
expiration = #1/1/2008#
      expiration = #8/27/2008 6:29:11 PM#
      expiration = "July 2, 2008"
      expiration = Today()
```

على فكرة ,الوظيفة Today() تعود بالتاريخ الحالي والوقت الحالي, بينما تعود الوظيفة Now() بالتاريخ الحالي,تستطيع أيضا استخلاص التاريخ الحالي باستدعاء الخاصية Today property لنوع بيانات التاريخ: Date.Today إشارة الدقة(pound sign) تخبر الفيجوال بيسك لان يخزن قيمة التاريخ للمتغير expiration تماما مثل ما تخبر علامتي الاقتباس(" ") فيجوال بيسك أن القيمة هي نص, تستطيع التاريخ كنص في متغير التاريخ .ولكن سيتم تحويله إلى التنسيق المناسب,إذا كان خيار التدقيق فعال Strict option is on لا تستطيع أن تخصص التاريخ باستخدام تنسيق التاريخ الطويل(كما هو الحال في العبارة الثالثة "July 2, 2008" = expiration)

تنسق التاريخ يتم تحديده بواسطة الإعداد الإقليمية the Regional Settings التي جدها في لوحة التحكم(Control Panel)ففي الولايات المتحدة التنسيق هو mm/DD/YY أما في باقي البلدان فان التنسيق dd/mm/yyا ذا أسندت تاريخ غير صحيح لمتغير الزمن مثل 23/04/2002 فان العبارة ستيتم وضع خط احمر تحتها وستظهر رسالة خطأ في نافذة قائمة المهامTask List window ووصف الخطأ هو أن ثابت الزمن ليس صحيح (Date constant is not valid) نوع البيانات الزمنية مرنة بشكل مفرط ,فالفيجوال بيسك يعرف كيف يعامل قيم التاريخ والوقت ,لذلك ليس عليك أن تكتب كود معقد لإتمام التحويلات الضرورية.لمعالجة الوقت والتاريخ استخدم عناصر النوع الزمني Date type والتي نوقشت بالتفصيل في الفصل الثالث عشر,أو وظائف التاريخ والوقت(date and time functions of VB 6) والتي ما تزال مدعومة في الفيجوال بيسك 2008 تستطيع أيضا تنفيذ عمليات رياضية بقيم التاريخ ,تدرك الفيجوال بيسك قصدك فن طرح الأزمنة من بعضها البعض وتقيم الفرق(الاختلاف)بشكل مناسب., والنتيجة ستكون كائن المدة الزمنية TIMESPAN object والتي تمثل الفترة الزمنية time interval . إذا نفذت العبارة التالية فان القيمة 1921.18:08:45.0468750 ستظهر في نافذة المخرجات immediate window:

```
Dim d1, d2 As Date
      d1 = Now
      d2 = #1/1/2004#
```

```
Debug.WriteLine(d1 - d2)
```

القيمة للكائن TIMESPAN تمثل فترة : 1921 يوم و18 ساعة و8 دقيقة و 45.0468750ثانية

معرفات نوع البيانات Data Type Identifiers

أخيرا تستطيع حذف الشرط As clause من عبارة Dim, بعد أن تنشئ متغيرات نوعية (مخصصة)بواسطة رموز تصريح المتغيرات variable declaration characters أو معرفات نوع البيانات data type identifiers, وهذه القيم الحرفية هي رموز خاصة والتي تزيلها إلى اسم المتغير للدلالة عنdenote نوع المتغير. لإنشاء متغير نصي تستطيع استخدام العبارة التالية: Dim myText$ اشارة الدولار تعني (تفيد signifies) متغير نصي . لاحظ أن اسم المتغير متضمن إشارة الدولار.وهي myText$, وليس myText .

لإنشاء متغيرات لأنواع معينة استخدم واحد من حروف التصريح الظاهرة في الجدول التالي (ليس جميع معرفات الأنواع موجود في هذا الجدول) استخدام هذه المعرفات لا يساعد في إنتاج كود أكثر قابلية للقراءة ولكنها تركة relics من الإصدارات القديمة جدا للبيسك.

أمثلة	نوع البيانات	الرمز
A$, messageText$	نصString	$
counter% , var%	عدد صحيح (Int32) Integer	%
population& , colorValue&	عدد صحيح طويل (Int64) Long	&
distance!	عدد كسري مفرد Single	!
ExactDistance#	عدد كسري مزدوج Double	#
Balance@	عدد عشريDecimal	@

خيارات التدقيق, التصريح , الاستنتاج The Strict, Explicit, and Infer Options

يزودك مترجم الفيجوال بيسك بثلاث خيارات والتي تحدد كيفية معالجة المتغيرات:

1- خيار التصريح Explicit option للدلالة فيما إذا كنت ستصرح عن جميع المتغيرات.

2- خيار التدقيق Strict option للدلالة فيما إذا جميع المتغيرات ستكون محددة النوع.

3- خيار الاستنتاجInfer option للدلالة فيما إذا المترجم سيحدد نوع المتغير من خلال قيمته.

لهذه الخيارات تأثير عميق على الطريقة التي تصرح وتستخدم فيها المتغيرات,وستفهم ما تعمله هذه الخيارات من خلال استكشاف هذه الإعدادات. وسوف تفهم أيضا بشكل أفضل كيف يعامل المترجم المتغيرات. من الموصى به أن تعمل على تفعيل(on)جميع هذه الخيارات ولكن مبرمجي البيسك القديمين قد لا يأخذون بهذه النصيحة.VB 2008 لا تتطلب التصريح عن المتغيرات, ولكن السلوك الافتراضي سيرمي باستثناء إذا حاولت استخدام متغير لم يتم التصريح عنه من قبل,إذا لم تصرح عن أسماء المتغيرات التي تظهر في كودك فان المحرر سيضع تحتها خط متعرج. تدل على ضبط خطأ, إذا حركت المؤشر فوق هذا الاسم الذي تحته خط سترى وصف للخطأ في صندوق أداة القائدة ToolTip box , لتغير السلوك الافتراضي عليك إدخال العبارة التالية في بداية الملف Option Explicit Off. وهذه العبارة يجب أن تظهر في أعلى الكود تماما وقبل أي حدث . وهي تؤثر على الكود الحالي فقط والتي تم كتابتها فيه وليس على المشاريع الأخرى والجديدة.وتستطيع تفعيلها بالإضافة إلى خيار التدقيق Strict لكامل المشروع من صفحة خصائص المشروع .اضغط يمين على اسم الحل في مستكشف الحل Solution Explorer واختار من القائمة المنسدلة خصائص Properties اختار اللوحة Compile tab وقم بتغير إعدادات الخيارات وفقا لذلك,وكذلك تستطيع الوصول إلى النافذة الافتراضية للبيسك من قائمة أدوات Tools menu اختار خيارات Options وعندما يظهر صندوق الحوار خيارات Options دد في اللوحة اليسارية Projects And Solutions ومنه انتقل إلى لوحة VB Defaults tab كما في الشكل(2،2): هنا تستطيع اعداد القيم الافتراضية لجميع الخيارات، وستبقى تستطيع تغير القيم الافتراضية لمشروع خاص من خلال صفحة خصائص المشروع project's Properties pages فطريقة عدم التصريح عن المتغيرات تم معالجتها في البيسك 2008 من خلال الخيارين التصريح Explicit والتدقيق Strict فخيار التصريح وهو فعال يطلب أن تكون جميع المتغيرات المستخدمة في الكود مصرح عنها مسبقا.بينما خيار التدقيق وهو فعال أيضا يتطلب أن تكون جميع المتغيرات محددة النوع,أي أن خيار التدقيق Strict option إذا كان فعال لا يسمح باستخدام النوع العام والذي يستطيع أن يخزن أي نوع من البيانات,القيمة الافتراضية لعبارة التصريح هي (ON) فعال وهي أيضا القيمة الموصى بها دائما,أما إذا لم تقم بتفعيل هذين الخيارين فانك تستطيع استخدام المتغيرات دون أن تصرح عنها وسيستخدم المترجم المتغير العام .

خيار التدقيق يكون غير فعال بشكل افتراضي فإذا تم تفعيله (ON)يجب عليك التصريح عن المتغيرات بنوع محدد من المتغيرات والمتغير سيقبل قيم من نفس النوع فقط ولكن مع خيار التدقيق غير فعال تستطيع مثلا استخدام متغير نصي والذي يحفظ عدد في حسابات عددية كما يلي : وستكون النتيجة هي: 12500

```
Dim a As String = "25000"
Debug.WriteLine((a) / 2)
```

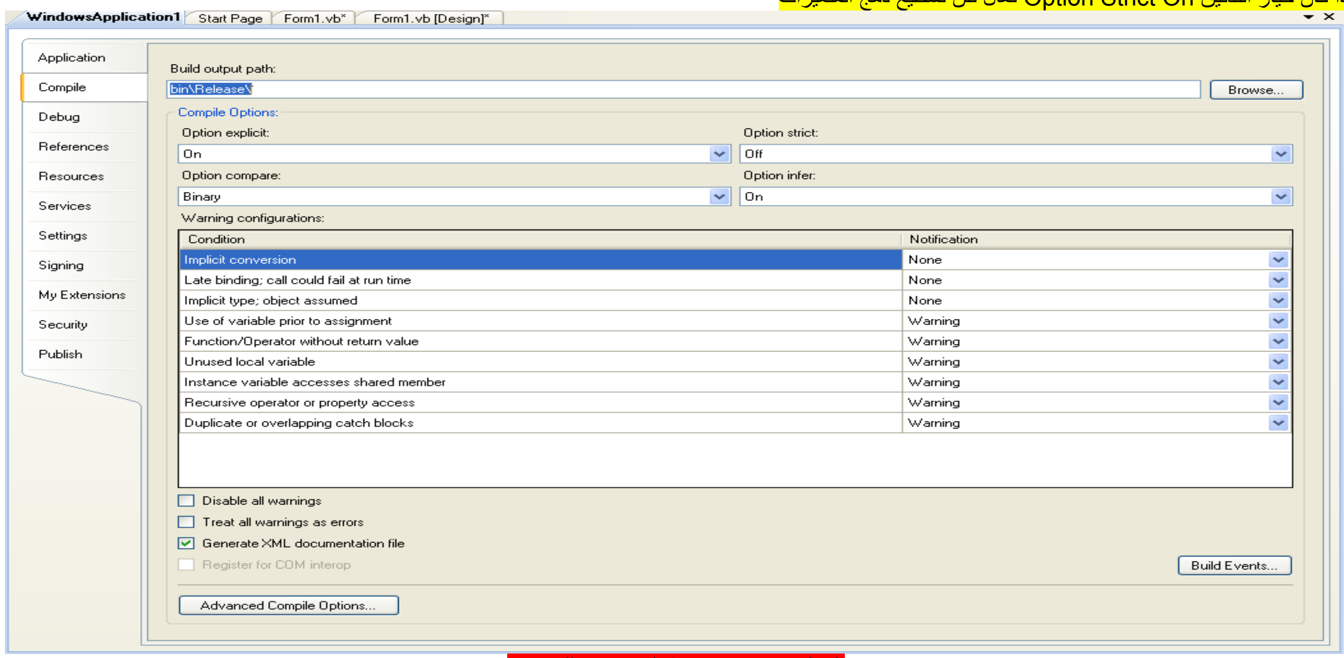
اما اذا تم تفعيل الخيار Strict option فيتوجب عليك كتابة العبارة السابقة كمايلي:

```
Dim a As String = "25000"
Debug.WriteLine(CDbl(a) / 2)
```

بطريقة مماثلة تستطيع استخدام المتغيرات العددية في الحسابات النصية :وستكون نتيجة العبارة التالية هي: 32.03 هذافي حال Option Strict off غير فعال

```
Dim a As Double = 31.03
a = a + "1"
Debug.WriteLine(a)
```

اما اذا كان خيار التدقيق Option Strict On فعال فلن تسطيع دمج المتغيرات



الشكل رقم(2،1)صفحة الخصائص للمشروع

Options

Environment
Projects and Solutions
General
VB Defaults
Text Editor
Windows Forms Designer
Device Tools

Default project settings:

Option Explicit: On
Option Strict: Off
Option Compare: Binary
Option Infer: On

Show all settings

OK Cancel

الشكل(2،2) خيارات البيسك

المتغيرات العامة(الكائنة) Object Variables

المتغيرات العمومية Variants بدون نوع بيانات ثابت ,كان الخبز والجبن لمبرمجي البيسك 6 وما قبلها , العموميات Variantsهي ضد المتغيرات المحددة النوع بشكل صارم strictly typed variables, فتستطيع تخزين أي نوع من البيانات من حرف مفرد إلى الكائن, إذا بدأت مع البيسك 2008 عليك أن تستخدم المتغيرات المحددة النوع , العموميات variants هو الجزء الرئيسي من تاريخ البيسك,العموميات أو المتغيرات الكائنية كانت أنواع البيانات الأكثر مرونة لأنها تستطيع أن تتكيف (accommodate)مع جميع الأنواع (فالمتغيرات التي يتم التصريح عنها ككائنات أو لا يتم التصريح عنها على الإطلاق) تعامل في الفيجوال بيسك تبعا لمحتوى المتغير الحالي,فإذا أسندت قيمة صحيحة integer لمتغير من ألكائني , فالبيسك سيعامله كعدد صحيح integer وهكذا, ولن اشرح في هذا النوع أكثر من ذلك,(فهي تستطيع تخزين جميع أنواع البيانات)

المتغيرات كـــكائنات Variables as Objects

المتغيرات في البيسك هي أكثر من كونها أسماء أو مكان لحفظ القيم, انها كينونات entities ذكية intelligent ولا تخزن فقط البيانات وإنما تعالج قيمها أيضا يمكن أن تكون قد سمعت بان المتغيرات في البيسك2008 هي كائنات واليك لماذا, المتغيرات التي تحفظ الزمن يتم التصريح عنها بمثل(أو بشكل مشابه لـــ as such) كما يلي:

Dim expiration As Date

ومن ثم تستطيع أن تسند قيمة تاريخ إلى المتغير expiration بعبارة مشابهة للتالية : #expiration = #1/1/2003 حتى الآن لا يوجد شيء غير عادي هذه هي الطريقة التي تستخدم فيها المتغيرات مع أية لغة أخرى بالإضافة إلى حفظ التاريخ, مهما يكن المتغير expiration يستطيع معالجة التواريخ, التعبير التالي سيعود بتاريخ جديد مع ثلاث سنوات للأمام three years ahead عن التاريخ الذي تم تخزينه في تعبير المتغير:

expiration.AddYears(3)

والتاريخ الجديد يمكن إسناده إلى متغير زمني أخر :

```
Dim newExpiration As Date
newExpiration = expiration.AddYears(3)
```

ان AddYears طريقة تعرف كيف تضيف عدد من السنين إلى متغير التاريخ ويوجد طرق أخرى مشابهة محجوزة لإضافة أشهر,أو أيام, وهكذا, بالإضافة إلى الطرق فان النوع تاريخ يعرض الخاصيات مثل خاصيات الأشهر والأيام Month and Day properties والتي تعود بشهر التاريخ وعدد الأيام على الترتيب,تماما مثل الخصائص والطرق للأدوات التي تضعها على الفورم,لإنشاء واجهة تطبيق المرئية application's visual interface. الطرق والخصائص methods and properties (المكونات members) لمتغير ما تعرض الدالة (الوظيفة)التي تبنى في فئة ممثلة المتغير نفسه.,

بدون دالة(وظيفة) البناء الداخلي (الذاتي)built-in functionality سيكون عليك كتابة بعض الكود الجاد لاستخراج الأشهر من متغير التاريخ. أو لإضافة عدد من الأيام, لتستكشف figure out فيما إذا قيمة حرفية هي حرف أبجدي letter أو رقم digit أو علامة ترقيم(punctuation symbol) الكثير من الدالات functionalityالتي تحتاجها في تطبيق ما والذي يعالج التاريخ, أو الأعداد, أو النصوص بشكل مسبق تم بناءها في المتغيرات نفسها,لا تدع المصطلحات terminology تخيفك , فكر في المتغيرات كحافظات للقيم وللتمكن من الوصول إلى دالاتها بتعابير مثل السابقة, ابدأ باستخدام المتغيرات كحافظات للقيم وعندما تريد أن تعالج هذه القيم فقط اكتب اسم المتغير متبوعا بنقطة لرؤية قائمة بالأعضاء التي تعرضها, في معظم الحالات ستكون قادر على استكشاف ما تفعله هذه الأعضاء بقراءة أسمائها فقط . لغات البرمجة تستطيع معاملة المتغيرات البسيطة بفعالية أكثر من الكائنات, فعدد integer صحيح ما يأخذ 2بايت(بشكل افتراضي الأعداد الصحيحة تمثل بالصحيح القصير(16)short) في الذاكرة والمترجم سيولد كود فعال جدا لمعالجة متغير صحيح integer variable(مثل إضافته إلى قيمة رقمية أخرى,مقارنته بعدد صحيح أخر, وهكذا) فإذا صرحت عن متغير صحيح واستخدمته في كودك as كمثل(كـــ)الفيجوال أستوديو لا تنشئ كائن لتمثيل هذه القيمة,بل تنشئ متغير جديد لتخزين الأعداد الصحيحة مثل البيسك القديم, بعد استدعاء واحد من طرق المتغير فان المترجم يصدر الكود لإنشاء الكائن الفعلي, هذه العملية تدعى البوكسينغ boxing (الصندقة)أو التغليف وتقدم القليل من البطء(التأخير)delay) والتي تعتبر غير مهمة(تافهة insignificant)مقارنة بالمعالجة المريحة convenience of manipulating للمتغير من خلال طرقه, كما رأيت لان إن المتغيرات هي كائنات, وهذه ليست مفاجئة ولكنه مبدأ غريب(odd)بالنسبة للمبرمجين الذين ليس لديهم خبرة في البرمجة كائنية التوجه object-oriented programming (فالكائنات كينونة تعرض بعض الدوال بوسائل الطرق والخصائص,. فأداة صندوق النص TextBox هي كائن وتوفر الخاصية Text التي تسمح لك بقراءة أو وضع نص على الأداة, أي اسم يتم إتباعه بنقطة واسم أخر يدل signifiesعلى كائن والاسم الأخر هو الخاصية أو الطريقة للكائن.)

Converting Variable Types

In many situations, you will need to convert variables from one type into another. Table 2.4 shows the methods of the Convert class that perform data-type conversions.

تحويل أنواع المتغيرات Converting Variable Types

تحتاج في بعض الحالات إلى تحويل المتغيرات من نوع إلى أخر الجدول التالي يعرض طرق فئة التحويل Convert class والتي تنجز تحويلات أنواع البيانات.

جدول طرق تحويل أنواع البيانات لفئة التحويل Data-Type Conversion Methods of the Convert Class

الطريقة **Method**	تحول معاملها النسبي إلى **Converts Its Argument To**
ToBoolean	Boolean
TOBYTE	Byte
TOCHAR	Unicode character
TODATETIME	Date
TODECIMA	Decimal
TODOUBLE	Double

TOINT16	Short Integer (2-byte integer, Int16)
TOINT32	Integer (4-byte integer, Int32)
TOINT64	Long (8-byte integer, Int64)
TOSBYTE	Signed Byte
TOSINGLE	Single
TOSTRING	String
TOUINT16	Unsigned Integer (2-byte integer, Int16
TOUINT32	Unsigned Integer (4-byte integer, Int32)
TOUINT64	Unsigned Long (8-byte integer, Int64)

بالإضافة إلى طرق فئة التحويل ,ما تزال تستطيع استخدام وظائف تحويل البيانات للفيجوال بيسك (()CInt لتحويل قيمة رقمية إلى قيمة صحيحة,()CDbl لتحويل القيم العددية الى قيمة عددية من النوع مزدوج Double , ()CSng لتحويل القيم العددية إلى مفرد, وهكذا) يمكن أن تلقي عليها نظرة في مستندات مايكروسوفت إذا كنت تكتب تطبيق جديد في VB 2008 استخدم فئة التحويل الجديدة للتحويل بين أنواع البيانات,لتحويل متغير تم التمهيد له كما يلي:

```
Dim A As Integer
```

إلى مزدوج استخدم الطريقة TODOUBLE لفئة التحويل:

```
Dim B As Double
B = Convert.ToDouble(A
```

افترض انك صرحت عن عددين صحيحين كما يلي:

```
Dim A As Integer, B As Integer
A = 23
B = 7
```

النتيجة للعملية A / B ستكون قيمة من النوع المزدوج وهي : 3.28571428571429

```
Debug.Write(A / B)
```

النتيجة قيمة مزدوجة والتي تزود بالدقة الأكبر المكنة ,إذا حاولت إسناد النتيجة إلى متغير لم يتم الإعلان عنه كمزدوج ,وخيار التدقيق فعالStrict option is on فان الفيجوال بيسك سيولد رسالة خطأ ,ولا نوع من أنواع البيانات يستطيع أن يقبل هذه القيمة بدون أن تفقد الدقة,لذا لتخزين النتيجة إلى متغير مفرد عليك تحويلها بشكل صريح بعبارة مثل التالية:

```
Convert.ToSingle(A / B)
```

تستطيع أيضا استخدام الوظيفة ()DirectCast لتحويل المتغير أو التعبير من نوع إلى أخر , الوظيفة()DirectCast مطابقة للوظيفة ()CType .دعنا نقول أن المتغير A تم التصريح عنه كنص ويحفظ القيمة 34.56 العبارة التالية تحول القيمة للمتغير A إلى قيمة عشرية وتستخدمها في الحسابات:

```
Dim A As String = "34.56"
Dim B As Double
B = DirectCast(A, Double) / 1.14
```

التحويل ضروري فقط إذا كان خيار التدقيق فعالStrict option is on ,ولكن من الجيد التدريب على انجاز تحويلاتك بشكل صريح.المقطع التالي يشرح ما يمكن أن يحدث فيما إذا اعتمد كودكrelies على التحويلات الضمنية implicit conversions .

التحويلات الواسعة والضيقة Widening and Narrowing Conversions

في بعض الحالات,سيقوم البيسك 2008 بتحويل أنواع البيانات بشكل أوتوماتيكي , ولكن ليس دائما ‘لنقول انك قد قمت بالتصريح عن متغيران وقمت بعملية التمهيد لهما,أحدهما صحيح والأخر مزدوج بالعبارات التالية:

```
Dim count As Integer = 99
Dim pi As Double = 3.1415926535897931
```

إذا كان خيار التدقيق غير فعالStrict option is off وأسندت المتغير piإلى count فان القيمة الجديدة للمتغيرcount ستكون 3 (القيمة المزدوجة تم تدويرها إلى قيمة صحيحة تبعا إلى نوع المتغير)على الرغم من أن هذا ما تريده في معظم الحالات,إنه إغفال (oversight)أو خطأ غير مقصود والذي يقود إلى نتيجة غير صحيحة .فإذا كان خيار التدقيق فعال Strict option is on وحاولت إتمام نفس الإسناد السابق ,فان المترجم سيولد رسالة خطأ ليدل على أنك لا تستطيع تحويل المزدوج إلى صحيح .نفس الرسالة هي آن خيار التدقيق لا يسمح بالتحويلات الضمنية من المزدوج إلى الصحيح .Option Strict disallows implicit conversions from Double to Integer, عندما يكون خيار التدقيق فعال Strict option is on فان البيسك سينجز التحويلات التي لا تنتج أي فقدان في الدقة ((accuracy (precision) أوmagnitude الأهمية(العظم) هذه التحويلات تدعى التحويلات الواسعة widening conversions عندما تسند قيمة صحيحة إلى مزدوج لن يتم فقدان دقة أو كمية magnitude.وهذا تحويل واسع لأنه يذهب من نوع أضيق إلى نوع أوسع,من ناحية أخرى عندما تسند قيمة مزدوجة إلى متغير قيمة صحيحة بعض الدقة سيتم فقدانها(الأرقام العشرية يجب أن تقطع أو تبتر truncated) هذا التحويل الضيق narrowing conversion لأنه يذهب من نوع البيانات الذي يمثل المجال الأوسع للقيم إلى نوع البيانات الذي يمكن أن يمثل المجال الأضيق للقيم .ولأنك أنت المبرمج ,وأنت المسئول ويمكن انك تريد على ما يبدو أن تقطع الدقة حيث انه لا يوجد حاجة لها. الجدول التالي يلخص التحويل الواسع الذي يقوم به الفيجوال بيسك بشكل أوتوماتيكي لك :

جدول التحويلات الواسعة للفيجوال بيسك 2008 VB 2008 Widening Conversions

نوع البيانات الأصلي**Original Data Type**	نوع البيانات الأوسع **Wider Data Type**
Any type	Object
Byte	Short, Integer, Long, Decimal, Single, Double
Short	Integer, Long, Decimal, Single, Double
Integer	Long, Decimal, Single, Double
Long	Decimal, Single, Double
Decimal	Single, Double
Single	Double
Char	String

تنسيق الأعداد Formatting Numbers

رأيت حتى الآن كيف تستخدم البيانات أنواع البيانات الأساسية للغة التنفيذ المشتركةCLR, جميع أنواع البيانات تعرض الطريقة ToString والتي تعود بقيمة المتغير(رقم أو تاريخ)كنص,لذلك يمكن استخدامها مع نصوص أخرى في كودك. الطريقة ToString تنسق الأعداد والتواريخ في عدة طرق ويمكن أن تكون واحدة من أكثر الطرق المعروفة التي تحتاج إليها,تستطيع استدعاء هذه الطريقة قيمة إلى نص بدون أي معامل نسبي كما فعلنا حتى الآن لتحويل أي.

الطريقة ToString تقبل معاملات نسبية اختيارية optional argument والتي تحدد كيف سيتم تنسيق القيمة كنص, مثلا.تستطيع تنسيق الأعداد كعملة بإضافة إليها سابقة إشارة مناسبة(مثلا إشارة الدولار,)وعرضها برقمين عشريين . وتستطيع عرض التاريخ في عدة هيئات.بعض التقارير تتطلب أن تكون الكمية السالبة مغلفة في قوسين enclosed in parentheses ,الطريقة ToString تسمح لك بعرض التاريخ بأي طريقة ترغبها,لاحظ أن ToString هي طريقة وليست خاصية وتعود بقيمة تستطيع أن تسندها إلى متغير نصي أو أن تمررها كمعامل نسبي لوظيفة مثل ()MsgBox ولكن القيمة الأصلية لا تتأثر. الطريقة ToString method تستطيع أيضا أن تنسق القيمة اذا ما استدعيت مع معاملات نسبية اختيارية,ToString(formatString) المعامل النسبي هوالمحدد للتنسيق (النص الذي يحدد التنسيق نفسه تماما الذي سيتم تطبيقه على المتغير)وهذا المعامل النسبي يمكن أن يكون رمز نوعي يطابق التنسيق المحدد مسبقا(نص التنسيق القياسي ,كما يسمى)أو نص من قيم حرفية والتي لها معنى خاص في عملية تنسيق القيم العددية (صورة تنسيق النص a picture format string) تستخدم النصوص بالتنسيق القياسي standard format strings لمعظم خيارات التنسيق المشتركة,وتستخدم صورة التنسيق للنص لتخصيص متطلبات تنسيق غير عادية,لتنسيق القيمة9959.95 ككمية دولار تستطيع استخدام العملة القياسية التالية:

```
Dim Amnt As Single = 9959.95
Dim strAmnt As String
strAmnt = Amnt.ToString("C")
```

أو استخدم صورة التنسيق العددي للنص التالي:

```
strAmnt = Amnt.ToString("$#,###.00")
```

كلا العبارتين سوف تنسق القيمة كـ $9,959.95 المعامل النسبي "C" في المثال الأول يعين عملة وينسق القيم العددية كـعملة,اذا كنت تستخدم نسخة ويندوز غير أمريكية non-U.S. version فان رمز العملة سوف يختلف تبعا للعملة التي تستخدمها في نسخة الويندوز خاصتك,استخدم أداة اللغة والإقليم في لوحة التحكم لتغير العملة إلى واحدة من العملات التي لديك في الويندوز,والكمية سيتم تنسيقها في إشارة العملة التي لديك,. تنسيق الشكلي للنص The picture format string مركبة من حروف ورموز of literals and characters والتي لها معنى خاص في التنسيق,إشارة الدولار ليس لها معنى خاص وستظهر كما هي , الرمز# هو حافظ رقمي (يحل مكانه رقم) وجميع الرموز # سيتم استبدالها بأرقام عددية,بداية من اليمين .اذا كان العدد لديه أرقام اقل من المحددة في النص فان الرموز الزائدة إلى اليسار سوف يتم تجاهلها,الفاصلة تخبر وتخبر وظيفة التنسيقFormat function بإدخال الفاصلة بين الألوف,.النقطة هي الفاصلة العشرية ,والتي يتم إتباعها بحافظتي رقم على خلاف الإشارة # فان الصفر 0 هو حافظة خاصة a special placeholder إذا كان لا يوجد أرقام كافية لجميع الأصفار التي حددتها فان الصفر سيظهر في مكان الأرقام المفقودة. فإذا كانت القيمة الابتدائية مثلا 9959.9 فان العبارة الأخيرة ستنسقها كالتالي: $9,959.90 , إذا استخدمت الحافظة# مكان الصفر فان النص الذي سيعود بواسطة طريقة التنسيق Format method لديه رقم عشري واحد فقط0

التنسيق العددي القياسي للنصوص Standard Numeric Format Strings

الطريقة ToString لنوع البيانات العددية تميز نصوص التنسيق العدية القياسية التي تظهر في الجدول التالي

حرف التنسيق	الوصف	الأمثلة
C or c	Currency(عملة)	(12345.67).ToString("C") ترجع بالقيمة : $12,345.67
D or d	Decimal (عشري)	(123456789).ToString("D") ترجع بالقيمة : 123456789. لا تعمل إلا مع القيم العددية الصحيحة
E or e	Scientific format(تنسيق علمي)	(12345.67).ToString("E") ترجع بالقيمة: 1.234567E + 004
F or f	Fixed-point format(تنسيق بفاصلة عائمة ثابتة)	(12345.67).ToString("F") ترجع بالقيمة: 12345.67
G or g	General format (تنسيق عام)	تعود بقيمة إما ذات فاصلة عائمة ثابتة أو تنسيق علمي
N or n	Number format (تنسيق عددي)	(12345.67).ToString("N") ترجع بالقيمة: 12,345.67
P or p	Percentage (نسبة مئوية)	(0.12345).ToString("N") ترجع بالقيمة 12,35%
R or r	Round-trip(خطوة التقريب)	(1 / 3).ToString("R") ترجع بالقيمة: 0.33333333333333331 إذا كان G specifier s المحدد G ستعود بقيمة بأرقام عشرية اقل 0.333333333333333
X or x	Hexadecimal format(التنسيق الست عشري)	250.ToString("X") ترجع بالقيمةFA

حروف التنسيق يمكن أن تتبع بأي عدد صحيح ,فإذا كان موجود فان القيم الصحيحة تخصص عدد أمكنة الرقم العشري التي تعرضها, الدقة الافتراضية هي رقمين عشريين, الحرف C ينسق النص الناتج عن الطريقة ToString ليعود بنص يمثل العدد كقيمة عملة ,العدد الصحيح الذي يتبع C يحدد عدد المراتب العشرية,التي سيتم عرضها,إذا لم تزود برقم فان الافتراضي أن تظهر رقمين عشرين بعد الفاصلة العشرية,افترض أن المتغير value تم التصريح عنها كـــعشري Decimal,وقيمته هي 5596 ومن ثم فان التعبير value.ToString("C") سيعود بالنص التالي : $5,596.00 إذا كانت قيمة المتغير 5596.4499 فان التعبير value.ToString("C3") سيعود بالنص. $5,596.450, لاحظ أن ليس جميع النصوص المنسقة تطبق على جميع أنواع البيانات, مثلا فقط القيمة الصحيحة يمكن تحويلها إلى التنسيق الست عشري ,ونص التنسيق D يعمل مع القيم الصحيحة فقط ,يوجد نصوص تنسيق و أرقام للتاريخ أيضا ,وتم مناقشتها في الفصل الثالث عشر من هذا الكتاب ,.

صورة التنسيق العددي للنصوص Picture Numeric Format Strings

إذا كانت حروف التنسيق المجدولة في الجدول السابق غير مناسبة للتحكم بالمظهر الذي تريده للقيمة العددية, تستطيع أن تقدم صورة التنسيق التي تخصصها أنت للنصوص, صورة تنسيق النصوص تحوي على رموز خاصة والتي تسمح لك بتنسيق قيمك تماما كما تريد,الجدول التالي يجدول رموز صور التنسيق,

حرف التنسيق	الوصف	الأمثلة
0	تعرض حافظة مكان لصفر	اذا كان العدد لديه أقل أرقام من الأصفار في التنسيق فالنتيجة ستكون بأصفار ليس لها معنى(أصفار على اليسار)
#	تعرض حافظ مكان لرقم	تعمل على تبديل الرمز فقط بأرقام لها معنى
.	الفاصلة العشرية	تعرض رمز النقطة
,	فاصل المجموعة	تفصل مجموعات الأرقام مثلا 1,000
%	رمز النسبة المئوية	rتعرض الرمز %
E + 0, E−0, e + 0, e−0	رمز الدليل(الأس) Exponent notation	تنسق المخرجات لرمز الأس(الدليل)
\	حرف أبجدي	تستخدم مع سلاسل التنسيق التقليدية مثل\n (newline)(سطر جديد)
" "	عارضات النص الحرفية Literal string Displays	تعرض أي نص ضمن علامتي الاقتباس أو الفاصلة العليا
;	فاصل مقطعي Section separator	تحدد مخرجات مختلفة اذا كانت القيمة العددية التي سيتم تنسيقها موجبة, أو سالبة,أو صفر

العبارات التالية ستطبع القيم التي بالخط العريض :

```
Dim Amount As Decimal = 42492.45
    Debug.WriteLine(Amount.ToString("$#,###.00"))
```

$42,492.45

```
    Amount = 0.2678
    Debug.WriteLine(Amount.ToString("0.000"))
```

0.268

```
    Amount = -24.95
    Debug.WriteLine(Amount.ToString("$#,###.00;($#,###.00)"))
```

($24.95)

أنواع البيانات المعرفة من قبل المستخدم User-Defined Data Types

في المقطع السابق استخدمنا المتغيرات لتخزين القيمة المستقلة ,وكمسألة واقعية,معظم المبرمجين يخزنون مجموعات من البيانات والتي تنتمي إلى أنواع مختلفة ,فمثلا برنامج لموازنة رصيد دفتر الشيكات يجب أن يخزن عدة قطع من المعلومات لكل شيك(رقم الشيك،الكمية,التاريخ,الشيك المدفوع,وهكذا)جميع هذه القطع من المعلومات ضرورية لمعالجة الشيك,وبشكل مثالي ideally يجب تخزين هذه المعلومات مع بعضها,تستطيع إنشاء أنواع بيانات مخصصة والتي تتركب من عدة قيم باستخدام التراكيبstructures, تركيب الفيجوال بيسك يسمح لك بجمع القيم المتعددة لأنواع البيانات الأساسية ومعالجتها جملة(as a whole)فمثلا كل شيك في تطبيق موازنة رصيد دفتر الشيكات يتم تخزينه في تركيب منفصل(أو سجل)كما في الشكل(3،2) عندما تستدعي شيك معطى,تحتاج إلى جميع المعلومات المخزنة في التركيب,

تركيب السجل

Check Number	Check Date	Check Amount	Check Paid To

مصفوفة السجلات

275	04/12/01	104.25	Gas Co.
276	04/12/01	48.76	Books
277	04/14/01	200.00	VISA
278	04/21/01	430.00	Rent

الشكل(2،3)رسم توضيحي(Pictorial) يمثل التركيب

لتعريف تركيب في الفيجوال بيسك, استخدم عبارة التركيب Structure والتي لها القوام التالي:

```
Structure structureName
      Dim variable1 As varType
      Dim variable2 As varType
        ...
      Dim variablen As varType
End Structure
```

يمكن لـ varType أن يكون أيا من أنواع البيانات المدعومة بواسطة لغة التنفيذ المشتركةCLR وعبارة Dim يمكن أن يتم تبديلها بأي من محددات(معدلات) الوصول access modifiers الخاصPrivate أو العام Public . من أجل التراكيب فقط فان عبارة Dim مكافئة لـــ Public,بعد هذا التصريح سيكون لديك نوع بيانات جديد يمكن أن تستخدمه في تطبيقك. اسم التركيب structureName يمكن أن يستخدم في أي مكان تستخدم فيه الأنواع الأساسية(Integers, Doubles, and so on),وتستطيع أن تعلن عن متغيرات من هذا النوع وتعالجها كما تعالج جميع المتغيرات الأخرى(مع بعض الزيادة القليلة في الكتابة). التصريح عن التركيب CheckRecord الموضح في الشكل(2،3) هو التالي:

```
Structure CheckRecord
      Dim CheckNumber As Integer
      Dim CheckDate As Date
      Dim CheckAmount As Single
      Dim CheckPaidTo As String
End Structure
```

يجب أن يظهر هذا التصريح خارج أي إجراء,فلا تستطيع التصريح عن تركيب Structure في إجراء فرعي أو وظيفة,. حالما يتم التصريح, فان التركيب CheckRecord يصبح نوع بيانات جديد لتطبيقك,وللتصريح عن متغير من هذا النوع ،استخدم عبارة مثل التالية:

```
Dim check1 As CheckRecord, check2 As CheckRecord
```

لتسند قيمة لواحد من هذه المتغيرات,يتوجب عليك أن تسند قيمة لكل واحد من مركباته بشكل منفصل(تدعى الحقول) والتي يمكن أن تصل أليها بدمج اسم المتغير واسم الحقل يفصل بينهما نقطة كما يلي:

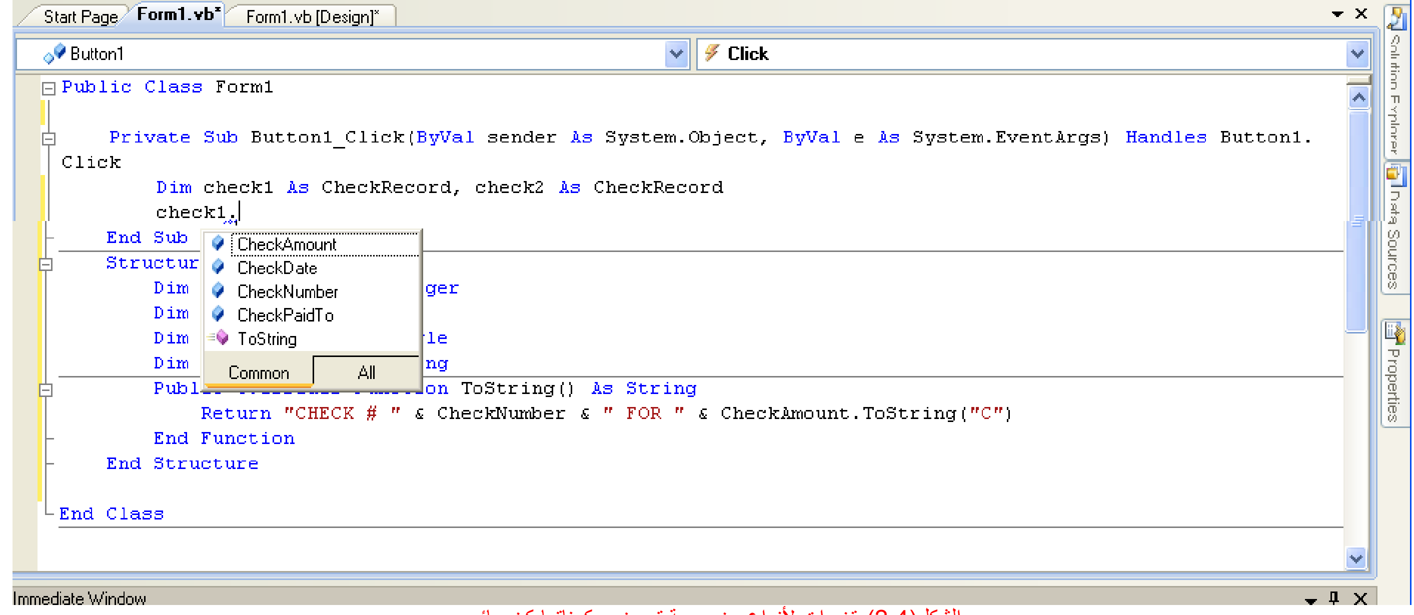
```
check1.CheckNumber = 275
```

عمليا حالما تكتب النقطة مجاور اسم المتغير, فان قائمة بجميع أعضاء التركيب ستظهر.كما هو مبين في الشكل(2،4),لاحظ أن التركيب يدعم العديد من المكونات بذاته حيث انه ليس عليك كتابة كود من أجل المكونات التالية:Equals, GetType و ToString ولكنها مكونات قياسية لأي كائن تركيب,وتستطيع استخدامها في كودك,فكلا الطريقتين GetType و ToString ستعود بنص مثل ProjectName.FormName(+CheckRecord) تسطيع ان تقدم معالجتك الخاصة للطريقة ToString والتي ستعود بنص أكثرأهمية:

```
Public Overrides Function ToString() As String
      Return "CHECK # " & CheckNumber & " FOR " & CheckAmount.ToString("C")
End Function
```

اكتب هذا التعريف للوظيفة ضمن التركيب أي كما يلي(إليك الكود كاملا):

```
Structure CheckRecord
      Dim CheckNumber As Integer
      Dim CheckDate As Date
      Dim CheckAmount As Single
      Dim CheckPaidTo As String
      Public Overrides Function ToString() As String
         Return "CHECK # " & CheckNumber & " FOR " & CheckAmount.ToString("C")
      End Function
End Structure
```



الشكل(2،4)متغيرات لأنواع مخصصة تعرض مكوناتها كخصائص

حالما تفهم أن التراكيب تشبه كثرا الكائنات التي تعرض حقولها كخصائص ومن ثم تعرض(تقدم)العديد من الأعضاء الخاصة بها هي,العبارة التالية تمهد للمتغير CheckRecord كما يلي:

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Dim check2 As CheckRecord
    check2.CheckNumber = 275
    check2.CheckDate = #9/12/2008#
    check2.CheckAmount = 104.25
    check2.CheckPaidTo = "Gas Co."
End Sub
```

تستطيع أيضا أن تنشئ مصفوفات من التراكيب بتصريح مثل التالي(المصفوفة ستناقش لاحقا في هذا الفصل):

```
Dim Checks(100) As CheckRecord
```

كل عنصر في هذه المصفوفة هو تركيب ويقوم بتثبت جميع الحقول لشيك معطى. للتمكن من الوصول إلى العنصر الثالث من المصفوفة استخدم الترميز التالي:

```
Checks(2).CheckNumber = 275
Checks(2).CheckDate = #9/12/2008#
Checks(2).CheckAmount = 104.25
Checks(2).CheckPaidTo = "Gas Co."
```

القيمة لاشيء The Nothing Value

تستخدم القيمة لاشيء مع المتغيرات الكائنية وتدل على أن المتغير لم يتم إسناد أي قيمة له بعد(لم يتم التمهيد له) إذا كنت تريد أن تعزل to disassociate متغير كائني ما عن الكائن الذي يمثله أسنده إلى لاشيء Nothing العبارات التالية تنشئ متغير كائني والذي يمثل الفرشاة استخدمه ومن ثم اعمل على تحريره:

```
Dim brush As SolidBrush
    brush = New SolidBrush(Color.Blue)
{ use brush object to draw with}
    brush = Nothing
```

العبارة الأولى تصرح عن المتغير brush, عند هذه النقطة المتغير brush لاشيء , العبارة الثانية تمهد للمتغير brush بواسطة المشيد المناسب(تم تمهيد الفرشاة إلى لون معين) بعد تنفيذ العبارة الثانية ,المتغير brush أصبح يمثل الآن كائن تستطيع الرسم به وباللون الأزرق. بعد استخدام الكائن لرسم شيء ما تستطيع تحريره وذلك بإسناد له القيمة لاشيء Nothing,اذا كنت تريد أن تستكشف فيما إذا المتغير ألكائني مسند له قيمة أم لا استخدم المعاملات Is و IsNot كما في المثال التالي:

```
Dim myPen As Pen
{ more statements here}
    If myPen Is Nothing Then
        myPen = New Pen(Color.Red)
    End If
```

المتغير myPen تم التمهيد له بواسطة constructorالمشيد New فقط اذا لم يتم التمهيد له من قبل, اذا أردت أن تحرر المتغير فيما بعد في كودك تستطيع إسناده إلى لاشيء بمعامل الإسناد(=) عندما تقارن الكائن بلاشيء Nothing, مهما يكن لا تستطيع استخدام معامل المساواة (=), عليك استخدام المعاملات Is و IsNot

Examining Variable Types

اختبار نوع المتغير Examining Variable Types

بالإضافة إلى إعداد أنواع المتغيرات و الوظائف للتحويل بين هذه الأنواع يقدم الفيجوال بيسك الطريقة GetType التي تعود بنص مع نوع المتغير: (Int32 صحيح 32,عشري Decimal,وهكذا)أي متغير يقدم هذه الطرق بشكل ألي,وتستطيع استدعاءها كما يلي:

```
Dim var As Double
    Debug.WriteLine("The variable's type is " & var.GetType.ToString)
```

العبارة في نافذة المخرجات ستكون: The variable's type is System.Double, يوجد أيضا المعامل operator GetType والذي يقبل as an argument كمعامل نسبي النوع ويعود بكائن النوع لنوع البيانات المحدد,الطريقة GetType method والمعامل GetType operator تستخدم على الأغلب في قوام If كما يلي:

```
If var.GetType() Is GetType(Double) Then
{ code to handle a Double value}
End If
```

لاحظ أن الكود لا يرجع بأسماء نوع البيانات مباشرة.عوضا عن ذلك يستخدم القيمة المعادة بواسطة المعامل GetType operator لاستخلاص نوع فئة المزدوج System.Double ومن ثم يقارن هذه القيمة بنوع المتغير بواسطة المعامل Is(أو IsNot)

هل هو عدد,نص, أو تاريخ؟ Is It a Number, String, or Date?

مجموعة أخرى من وظائف الفيجوال بيسك والتي تعود بأنواع بيانات المتغيرات ,ولكن ليس النوع بالضبط,فهي تعود بقيمة صح/خطأ, لتدل على أن المتغير لديه قيمة عددية أو تاريخ أو مصفوفة, أم لا.,. الوظائف التالية تستخدم للتحقق من إدخال المستخدم,بالإضافة إلى البيانات المخزنة في الملفات,قبل أن تقوم بمعالجتها,

1- هل هو عدد IsNumeric ()

تعود بصح اذا كان معاملها النسبي هو عدد(قصير,صحيح وطويل,مفرد,مزدوج،وعشري)استخدم هذه الوظيفة لتحديد فيما إذا المتغير لديه قيمة عددية قبل تمريره إلى الإجراء الذي يقبل قيمة عددية أو قبل معالجته كعدد, العبارة التالية تبقى تطالب المستخدم من خلال صندوق الإدخال InputBox بقيمة عددية ,ويتوجب على المستخدم أن يدخل قيمة عددية أو ينقر على زر إلغاء للخروج ,طالما أن المستخدم يدخل قيم غير عددية فان صندوق الإدخال سيبقى يبرز للمستخدم ويطالبه بإدخال قيمة عددية:

```
Dim strAge As String = ""
    Dim Age As Integer
    While Not IsNumeric(strAge)
      strAge = InputBox("Please enter your age")
    End While
    Age = Convert.ToInt16(strAge
```

المتغير strAge تم التمهيد له بقيمة non-numeric value غير عددية لذلك فان الحلقة End While loop. . .While سيتم تنفيذها مرة على الأقل .

هل هو تاريخ IsDate

IsDate تعود بصح اذا كان المعامل النسبي هو تاريخ صحيح (أو وقت) التعبير التالي يعود بصح لأنها تمثل جميعها تاريخ صحيح:

```
IsDate(#10/12/2010#)
IsDate("10/12/2010")
IsDate("October 12, 2010")
```

اذا تعبير التاريخ يتضمن اسم اليوم كما في التعبير التالي , فان الوظيفة IsDate ستعود بخطأ:

```
IsDate("Sat. October 12, 2010") ' FALSE
```

هل هو مصفوفة IsArray()

تعود بصح اذا كان معاملها النسبي مصفوفة.

لماذا التصريح عن المتغيرات؟ Why Declare Variables?

لا تجبر أبداً الفيجوال بيسك على التصريح عن المتغيرات (ولا تزال كذلك) وهذا شيء جيد بالنسبة للمبتدئين في البرمجة , ولكن عندما تكتب تطبيقات ضخمة,ستكتشف أن التصريح عن المتغيرات ضروريا,سوف يساعدك على كتابة كود واضح وفعال بشكل قوي جدا,ويبسط أيضا عملية ترجمة الكود وتصحيح الأخطاء,فالتصريح عن المتغيرات يخلص الكود المصدر من معظم ومجمل الأخطاء المعروفة و الغير ضرورية,دعنا نتفحص التأثيرات الجانبية لاستخدام المتغيرات غير المصرح عنها في تطبيقك,لتكون قادر على عدم التصريح عن المتغيرات عليك أن تعمل على اعداد خيار التصريح بحيث يصبح غير فعال Explicit option to Off,لنفرض انك تستخدم العبارات التالية .تحويل اليورو إلى الدولار الأمريكي:

```
Euro2USD = 1.462
USDollars = amount * Euro2USD
```

في المرة الأولى التي يشير فيها الكود إلى اسم المتغير Euro2USD يعمل البيسك على إنشاء متغير جديد ومن ثم يستعمله كما لو انه تم التصريح عنه.,افترض أن المتغير Euro2USD يظهر في عدة أماكن في تطبيقك ,اذا ما كتبت Euro2UDS في واحد من هذه الأماكن والبرنامج لا يعمل على إجبار التصريح عن المتغيرات فان المترجم سيحمل متغير جديد ,ويسند له القيمة صفر ,ومن ثم يستخدمه.. أي كمية تم تحويلها بواسطة المتغيرEuro2UDS ستكون صفر,اذا كان التطبيق يجبر التصريح عن المتغيرات ,فان المترجم سيتذمر complain (من) أن المتغير Euro2UDS لم يتم التصريح عنه)وسوف تضبط الخطأ حالا وأنت في محرر الكود .

مجال المتغير A Variable's Scope

بالإضافة إلى النوع للمتغيرات أيضا مجال للرؤية,فالمجال(الرؤية)للمتغير هو المقطع من التطبيق الذي يمكنه رؤية ومعالجة المتغير . فإذا ما تم التصريح عن متغير ما ضمن إجراء فقط الكود في الإجراء المحدد يتمكن من الوصول إلى المتغير ,وهذا المتغير غير موجود لباقي التطبيق ,عندما يتم تحديد مجال المتغير ضمن إجراء فانه يدعى محلي(local)افترض انك تعمل على كتابة الكود لحدث النقر على الزر لحساب مجموع جميع الأعداد في المجال 0 إلى 100 ,واحدة من المعالجات الممكنة يوضحها الكود التالي :

```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)   Handles Button1.Click
    Dim i As Integer
    Dim Sum As Integer
    For i = 0 To 100 Step 2
      Sum = Sum + i
    Next
    MsgBox("The sum is " & Sum.ToString)

 End Sub
```

المتغير i والمتغيرSum هي متغيرات محلية للإجراء Button1_Click,اذا حاولت أن تضع قيمة المتغيرSum من ضمن إجراء آخر فان البيسك سيتذمر(complain)من أن المتغير لم يتم الإعلان عنه,(أو اذا جعلت خيار التصريح عن المتغيرات غير فعال Explicit option) سيحمل متغير أخر Sum وسيمهد له بإسناد القيمة صفر له,ومن ثم يستخدمه,ولكن هذا لن يؤثر على المتغير Sum الذي في الإجراءButton1_Click ويقال عن المتغيرSum ان لديه مجال على مستوى الإجراء:procedure-level scope أي انه مرئي ضمن الإجراء وغير مرئي خارج هذا الإجراء.. في بعض الأحيان تحتاج إلى استخدام المتغير بحدود مجال محدد بحيث يكون متاح إلى جميع الإجراءات ضمن ملف صغير هذا المتغير والذي تم التصريح عنه خارج أي إجراء يقال أن لديه مجال على مستوى الموديل(module-level scope) بشكل مبدأي تستطيع التصريح عن جميع المتغيرات خارج الإجراء الذي يستخدمهم,ولكن هذا سيقود إلى مشكلة ., كل متغير في الملف سيستطيع الوصول إلى أي متغير ,وتحتاج إلى أن تكون حذر بشكل كبير بان لا تغير القيمة للمتغير بدون سبب مقنع ,فالمتغيرات التي تحتاج إليها فقط في إجراء(مثل حلقة العداد loop counters) يجب ان يتم التصريح عنها في الإجراء.نوع أخر للمجال وهو المجال على مستوى البلوك(المتغيرات على مستوى الكتل) ,المتغيرات المقدمة في البلوك مثل عبارة if أو الحلقة هي محلية في البلوك وغير مرئية خارج البلوك. . دعنا نعدل مقطع الكود السابق بحيث يحسب مجموع المربعات, لتنفيذ الحساب ,نقوم أولا بحساب مربع كل قيمة ومن ثم نجمع المربعات,مربع كل قيمة يتم تخزينه في متغير والذي لن يتم استخدامه خارج الحلقة, لذلك نستطيع أن نعرف المتغير sqrValue في قوام الحلقة ونجعله محلي لهذه الحلقة المحددة كما هو الحال في القائمة التالية:

```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)  Handles Button1.Click
    Dim i, Sum As Integer
    For i = 0 To 100 Step 2
      Dim sqrValue As Integer
      sqrValue = i * i
      Sum = Sum + sqrValue
    Next
    MsgBox("The sum of the squares is " & Sum)
  End Sub
End Class
```

المتغير sqrValue غير مرئي خارج بلوك حلقة For. . .Next loop اذا حاولت استخدامه قبل عبارة For أو بعد العبارة Next سيرمي الفيجوال بيسك استثناء المتغيرsqrValue يحفظ قيمتهبين الدوران المتغيرات على مستوى البلوك لا يتم التمهيد لها في كل دوران. وحتى لو كان يوجد عبارة Dim في الحلقة .أخيرا في بعض الحالات فان التطبيق الكلي يجب أن يتمكن من الوصول إلى متغير معين,في هذه الحالة فان المتغير يجب أن يتم الإعلان عنه كمتغير عام Public والمتغيرات العامة لديها مجال عامglobal scope فهي مرئية من أي جزء في التطبيق,للإعلان عن متغير عام استخدم العبارةPublic مكان العبارة Dim وأكثر من ذاك لا تستطيع الإعلان عن متغير عام ضمن الإجراء,اذا كان لديك عدة نماذج في تطبيقك وتريد من الكود في أحد النماذج ان يرى متغير محدد في نموذج أخر استخدم محدد الوصولPublic modifier

الكلمة المحجوزة Public تجعل المتغير متاح ليس فقط من كامل المشروع ,إنما أيضا لمشاريع أخرى(بعبارة أخرى,متاح لجميع الإجراءات في لأي موديل في المشروع)المتغيرات التي تريد استخدامها على كامل مشروعك,ولكن لا تصبح متاحة للمشاريع الأخرى والتي تستعمل هذا المشروع كمرجع(reference) سيتم التصريح عنها كـــصديق Friend لذلك لماذا نحتاج إلى العديد من أنواع المجالات,؟ سوف تعمل على تطوير فهم أفضل للمجال وأي نوع من المجال يجب أن يتم استخدامه لكل متغير كلما انخرطت في مشاريع ضخمة, بشكل عام ,عليك المحاولة لتحديد مجال متغيراتك قدر المستطاع,اذا تم التصريح عن جميع المتغيرات ضمن الإجراءات,تستطيع استخدام نفس الاسم لتخزين قيمة مؤقتة,في كل إجراء وتكون واثق من أن متغيرا ت إجراء واحد لن تتداخل مع الإجراءات الأخرى,حتى ولو استخدمت الاسم نفسه,

عمر المتغير A Variable's Lifetime

بالإضافة إلى النوع والمجال للمتغيرات عمرlifetime والذي هو الفترة التي تحتفظ retain هذه المتغيرات بقيمتها,المتغيرات المصرح عنها كـــــPublic تكون موجودة بعمر التطبيق ,المتغيرات المحلية التي ضمن الإجراءات بعبارة Dim أو Private حياتها محددة بالإجراء,عندما ينتهي الإجراء,المتغير المحلي يتوقع عن الوجود cease to exist و الذاكرة المقسومة allocated memoryتعود إلى النظام ,بالطبع ,نفس الإجراء يمكن استدعاءه مرة أخرى.في هذه الحالة المتغيرات المحلية يتم إعادة إنشاءها ويتم إسناد القيم لها من جديد,اذا ما استدعى إجراء , إجراء أخر,فان متغيراته المحلية يتم حفظها بينما يكون الإجراء المدعو مشغل.,تستطيع أيضا أن تجبر المتغيرات المحلية على أن تحفظ قيمتها بين استدعاءات الإجراء between procedure calls باستخدام الكلمة المحجوزة Static افترض أن مستخدم تطبيقك يستطيع إدخال قيمة عددية في أي وقت,واحدة من المهمات المنجزة بواسطة التطبيق,هي حساب معدل القيم العددية,بدل إضافة جميع القيم في كل مرة يقوم المستخدم بإضافة قيمة جديدة ويقسمها على العداد تستطيع حفظ الكلي totalيعمل ,بواسطة الوظيفة() RunningAvg كما في الكود التالي:

```
Function RunningAvg(ByVal newValue As Double) As Double
    CurrentTotal = CurrentTotal + newValue
    TotalItems = TotalItems + 1
    RunningAvg = CurrentTotal / TotalItems
End Function
```

عليك التصريح عن المتغيرات CurrentTotal وTotalItems خارج الوظيفة لذلك فان قيمها ستيتم حفظها بين الاستدعاءات,بشكل اختياري تستطيع التصريح عنهم في الوظيفة باستخدام الكلمة المحجوزة Static keyword كما في الكود التالي :

```
Function RunningAvg(ByVal newValue As Double) As Double
    Static CurrentTotal As Double
    Static TotalItems As Integer
    CurrentTotal = CurrentTotal + newValue
    TotalItems = TotalItems + 1
    RunningAvg = CurrentTotal / TotalItems
End Function
```

الفائدة من استخدام المتغيرات المحلية هو أنها تساعدك على تقليل عدد المتغيرات الكلي في التطبيق,كل ما تحتاج إليه هو تشغيل وظيفة المتوسط average والتي تزود بها الوظيفة () RunningAvg بدون جعل متغيراتها مرئية إلى باقي التطبيق ,وهكذا لا تخاطر بتغير قيمة المتغيرات من ضمن إجراءات أخرى ,المتغيرات المصرح عنها في الموديلmodule خارج أي إجراء تأخذ تأثيرها procedure take effect عندما يتم تحميل الفورم وتتوقف عن الوجود عندما لا يتم تحميل الفورم.اذا تم تحميل الفورم مرة أخرى يتم التمهيد لمتغيراتها كما لو أنها تم تحميلها للمرة الأولى.,

يتم التمهيد للمتغيرات عندما يتم التصريح عنها تبعا لنوعها,التغيرات العددية يمهد لها إلى صفر والمتغيرات النصية يمهد لها إلى نص فارغ blank string ,والمتغيرات الكائنة إلى لاشيء.

الثوابت Constants

بعض المتغيرات لا تغير القيمة خلا تنفيذ البرنامج ,هذه المتغيرات هي ثوابت والتي يمكن أن تظهر عدة مرات في كودك,مثلا اذا كان البرنامج يقوم بعمل حسابات رياضية,فان قيمة الـ pi (3.14159. . .) يمكن أن تظهر عدة مرات,فبدلا من كتابة القيمة 3.14159 مرة بعد مرة ,تستطيع أن تعرف متغير كثابتconstant وتسميه pi(π) , وتستخدم اسمه في كودك , العبارة التالية أسهل في الفهم

```
circumference = 2 * pi * radius
```

من العبارة التالية:

```
circumference = 2 * 3.14159 * radius
```

يمكنك أن تعلن عن pi كمتغير ولكن الثابت هو المفضل ,لسببين:

1- الثوابت لا تغير القيمة :للامان في المستقبل . بعد أن يتم التصريح عن الثابت ,لا تستطيع تغير قيمته في العبارات اللاحقة,لذا عليك أن تكون واثق من القيمة المخصصة في تصريح الثابت ستأخذ تأثيرها في كامل البرنامج.

2- المتغيرات يتم معالجتها بشكل أسرع من المتغيرات .عندما يتم تشغيل البرنامج فان قيم الثوابت لا يتم البحث عنها.,يستبدل المترجم أسماء الثوابت بقيمها,وبالتالي يتم تنفيذ البرنامج بشكل أسرع .

الطريقة التي تصرح بها عن الثوابت مشابهة تماما لطريقة التصريح عن المتغيرات,ما عدا انك تستخدم الكلمة المحجوزة Const بالإضافة إلى التزويد باسم الثابت,ويجب عليك أيضا أن تقدم القيمة .كما يلي :

```
Const constantname As type = value
```

للثوابت أيضا مجال وتستطيع استخدام الكلمات المحجوزة Private وPublic فالثابت pi مثلا . يصرح عنه عادة في الموديل كعام(module as Public).ولذلك كل إجراء يمكن أن يصل إليه .

```
Public Const pi As Double = 3.14159265358979
```

اسم الثابت ,متبوعا بنفس القاعدة لاسم المتغيرات. قيمة الثابت هي قيمة حرفية,أو تعبير بسيط مكون من قيمة عددية, أو ثابت نصي ومعاملات, على فكرة, لا تستطيع استخدام الوظائف في التصريح عن الثوابت فالتعريف التالي غير مقبول:(Const Pi = Math.Log(Math.Pi ,القيمة الخاصة التي استخدمتها في هذا المثال لا تحتاج إلى تخزينها في ثابت, استخدم الــ pi أحد مكونات الفئة Math class للرياضيات بدلا عن ذلك(Math.PI) . يمكن للثوابت ان تكون نصوص,مثل التالي:

```
Const ExpDate = #31/12/1997#
Const ValidKey = "A567dfe"
```

المصفوفات Arrays

المصفوفة هي تركيب قياسي لتخزين البيانات في أي لغة برمجة ,ببينما المتغيرات المستقلة تستطيع أن كينونات, مثل عدد مفرد ,تاريخ واحد,أو نص مفرد, المصفوفات تستطيع أن تثبت مجموعات من البيانات في نفس النوع(مجموعة اعداد,سلسلة تواريخ,وهكذا). للمصفوفة اسم .كما تفعل مع المتغيرات وقيم يتم تخزينها فيها ويمكن الوصول إليها من خلال فهرس رقم الخانة index,مثلا تستطيع استخدام المتغير Salary لتخزين راتب شخص:

```
Salary = 34000
```

ماذا اذا كنت ترغب في تخزين رواتب 16 مستخدم؟ تستطيع إما أن تصرح عن 16 متغير Salary1 و Salary2 وهكذا حتى Salary16 أو التصريح عن مصفوفة ذات 16 عنصر .المصفوفة مشابهة للمتغير:لها اسم وقيم متعددة. وكل قيمة يتم تعريفها بواسطة فهرس(قيمة عددية صحيحة) والذي يتبع اسم المصفوفة ضمن الأقواس.كل قيمة مختلفة هي عنصر من المصفوفة.اذا كانت مصفوفة Salaries الرواتب تثبت الرواتب لـ16 مستخدم ,العنصر (Salaries(0 يثبت راتب الموظف الأول ,العنصر (Salaries(1 يثبت راتب الموظف الثاني ,وهكذا حتى العنصر (Salaries(15

التصريح عن المصفوفات Declaring Arrays

بشكل غير مشابه للمتغيرات البسيطة , المصفوفات يجب أن يتم الإعلان عنها بالعبارة Dim(أو عام Public) متبوعة باسم المصفوفة وفهرس العنصر(الخانة) الأخيرة للمصفوفة ضمن أقواس خذ المثال التالي: Dim Salary(15) As Integer هذه مصفوفة تحوي على 16 عنصر(خانة) من(0 إلى 15) اسمها Salary (راتب), ونوع بياناتها أعداد صحيحة.فــ (Salary(0 هو راتب الموظف الأول,و(Salary(1 راتب الموظف الثاني ,وهكذا . كل ما عليك فعله هو أن تتذكر :ما العنصر الذي يطابق كل راتب,ولكن حتى هذه البيانات يمكن معالجتها بواسطة مصفوفة أخرى,لفعل هذا ,عليك الإعلان عن مصفوفة أخرى من 16 عنصر(خانة).

```
Dim Names(15) As String
```

ومن ثم اسند القيم لكلا المصفوفتين :

```
Dim Names(15) As String
Dim salary(15) As Integer
    Names(0) = "Joe Doe"
    Salary(0) = 34000
    Names(1) = "Beth York"
    Salary(1) = 62000
```

```
        ...
    Names(15) = "Peter Smack"
    Salary(15) = 10300
```

هذا التركيب أكثر إحكام وراحة من لاستحواذ على كود بمنتهى الصعوبة يحفظ أسماء ورواتب الموظفين في متغيرات. جميع عناصر المصفوفة لها نفس النوع من البيانات,وبالطبع كذلك,عندما يكون نوع البيانات هو كائن , فان العناصر المستقلة تستطيع أن تحوي أنواع مختلفة من البيانات(كائنات , نصوص,اعداد,وهكذا). المصفوفات غير محددة بأنواع البيانات الأساسية,يمكن الإعلان عن مصفوفات تستطيع أن تحفظ أي نوع من البيانات بما فيها الكائنات فالمصفوفة التالية تحفظ الألوان التي تستطيع استخدامها فيما بعد في الكود كمعاملات نسبية لوظائف (دالات متنوعة) والتي ترسم الأشكال:

```
Dim colors(2) As Color
    colors(0) = Color.BurlyWood
    colors(1) = Color.AliceBlue
    colors(2) = Color.Sienna
```

فئة اللون تمثل الألوان ومن الخاصيات التي تقدمها هي أسماء الألوان التي تميزها,.التقنية الأفضل لتخزين الأسماء والرواتب هو إنشاء تركيب ومن ثم الإعلان عن مصفوفة و من هذا النوع كما في التركيب التالي الذي يثبت الأسماء والرواتب:.

```
Structure Employee
    Dim Name As String
    Dim Salary As Decimal
End Structure
```

ادخل التصريح السابق في ملف كود الفورم,وخارج أي إجراء ومن ثم صرح عن المصفوفة في إجراء كما يلي:

```
Dim Emps(15) As Employee
```

كل عنصر في المصفوفة يقدم حقلان ,وتستطيع أن تسند القيم لها باستخدام عبارة مثل التالية:

```
Emps(2).Name = "Beth York"
Emps(2).Salary = 62000
```

الفائدة من استخدام مصفوفة التركيب بدلا من المصفوفات المتعددة في ان المعلومات المرتبطة ستيتم دائما إيجادها تحت نفس الفهرس,فالكود أكثر إحكاما ولا تحتاج إلى حفظ العديد من المصفوفات.

التمهيد(اسناد قيمة اولية)للمصفوفات Initializing Arrays

كما تستطيع ان تمهد للمتغيرات في نفس الخط الذي تصرح فيه عنهم.تستطيع ايضا التمهيد للمصفوفات ,بالمشيد constructor التالي(ممهد المصفوفة array initializer عند استدعائها):

```
Dim arrayname() As type = {entry0, entry1, ... entryN}
```

اليك مثال الذي يعمل على التمهيد لمصفوفة نصية:

```
Dim Names() As String = {"Joe Doe", "Peter Smack"}
```

هذه العبارة مكافئة للعبارة التالية والتي تصرح عن مصفوفة ذات عنصرين ومن ثم تضع قيم فيهم :

```
Dim Names(1) As String
Names(0) = "Joe Doe"
Names(1) = "Peter Smack
```

عدد العناصر في الأقواس المجعدة curly brackets التي تتبع التصريح عن المصفوفة تحدد أبعاد المصفوفة ,ولا تستطيع اضافة عناصر جديدة للمصفوفة بدون إعادة تحجيمها,إذا احتجت الى اعادة تحجيم المصفوفة في كودك بشكل ديناميكي, عليك استخدام العبارة ReDim سيتم شرحها فيما بعد, في هذا الفصل .مهما يكن تستطيع ان تغير القيمة في العناصر الموجودة كما تريد at will وكما تعمل مع أي مصفوفة أخرى.

حدود المصفوفة Array Limits

العنصر الأول للمصفوفة لديه الفهرس صفر(index 0) ,والعدد الذي يظهر بين الأقواس في عبارة Dim أقل بواحد من السعة الكلية(حجم المصفوفة) للمصفوفة وهو الحد الأعلى(upper bound) (أو النهاية العليا upper limit)للمصفوفة, وفهرس العنصر الأخير للمصفوفة هو الحد الأعلى لها(upper bound)ويعطى بالطريقة GetUpperBound والتي تقبل كمعامل نسبي بعد المصفوفة وتعود بالحد الأعلى لهذا البعد,والمصفوفات التي رأيناها حتى الأن هي مصفوفات ذات بعد مفرد(one-dimensional) وبالتالي المعامل النسبي الذي يجب ان يمرر الى الطريقة(اقرا الحد الأعلى GetUpperBound)هو القيمة صفر 0 ,.عدد عناصر المصفوفة الكلي (طول المصفوفة)يعطى بالطريقة(إقرا الطول GetLength) والتي تقبل ايضا البعد كمعامل نسبي, الحد الأعلى للمصفوفة التالية هو 19 وسعتها المصفوفة (حجم المصفوفة هو 20 عنصر)

```
Dim Names(19) As Integer
```

العنصر الأول هو ,Names(0) والعنصر الأخير هوNames(19) فاذا ما نفذت العبارات التالية ستظهر لك النتائج التي بالخط العريض في نافذة المخرجات:

```
Debug.WriteLine(Names.GetLowerBound(0))  0  النتيجةستكون
Debug.WriteLine(Names.GetUpperBound(0))  19  النتيجة ستكون
```

لتسند قيمة لأول وأخر عنصر من المصفوفة Names إستخدم العبارات التالية

```
Names(0) = "First entry"
Names(19) = "Last entry"
```

اذا اردت ان تعمل دوران على عناصر المصفوفة استخدم حلقة مثل التالي:

```
Dim i As Integer, myArray(19) As Integer
    For i = 0 To myArray.GetUpperBound(0)
        myArray(i) = i * 1000
    Next
```

العدد الحقيقي للعناصر في أي مصفوفة يعطى بالتعبير التالي:

```
myArray.GetUpperBound(0) + 1
```

تستطيع ايضا استخدام خاصية طول المصفوفة Length لاستخلاص عدد العناصر(احصاء العناصر)فالعبارة التالية تطبع عدد العناصر التي في المصفوفة myArray في نافذة المخرجات(ملاحظة :عندما اقول نافذة المخرجات اقصد النافذة المباشرة immediate window)

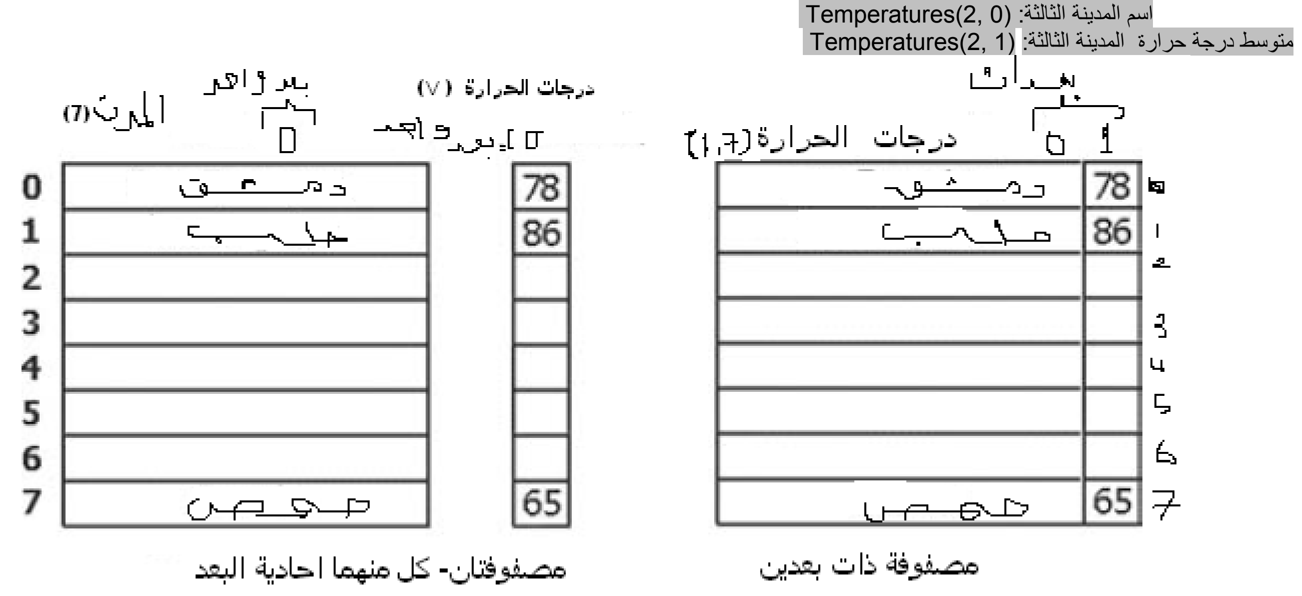
```
Debug.WriteLine(myArray.Length)
```

هل ما تزال مشوش بشان مكيدة الفهرس صفر zero-indexing scheme وعدد العناصر وفهرس للعنصر الاخير في المصفوفة؟اذا كان الامر كذلك .تستطيع جعل المصفوفة أكبر نوعا ما وتتجاهل العنصر الاول,فقط كم واثق من انك لن تستخدم العنصر صفر في كودك,فلا تخزن قيمة في العنصر Array(0) وبالتالي تستطيع تجاهل هذا العنصر,فللحصول على عشرين 20 عنصر ,اعلن عن مصفوفة ذات 21 عنصركما يلي: Dim MyArray(20) As type ومن ثم تجاهل العنصر الاول.

المصفوفات المتعددة الابعاد Multidimensional Arrays

المصفوفات ذات البعد الواحد كالتي عرضناها حتى الان جيدة لتخزين سلاسل بيانات طويلة و ذات بعد واحد (كالاسماء ,درجات الحرارة)ولكن كيف ستخزن قائمة من المدن مع متوسط درجة حرارتها في مصفوفة ؟ أو الأسماء والنقاط المكسوبة في لعبة,أو السنوات و الربح(الفائدة)؟ أو البيانات مع أكثر من بعدين(data with more than two dimensions)؟ مثل المنتجات والاسعار والوحدات المخزنة(المتبقية في المخازن) سوف تخزن سلاسل من البيانات المتعددة الأبعاد,. تستطيع ان تخزن نفس البيانات بشكل اكثر ملائمّة في مصفوفة متعددة الأبعاد بقدر ما تحتاج(أي بكلام اخر عدد ابعاد المصفوفة تحددها الحاجة لهذه الابعاد اوالبيانات المتنوعة التي ستخزنها في هذه المصفوفة),الشكل(5،2)يبين لك اثنين من المصفوفات الأحادية البعد one-dimensional arrays واحدة منهم مع اسماء المدن والأخرى مع درجات الحرارة, واسم المدينة الثالثة سيكون City(2), ودرجة حرارتها ستكون Temperature(2). المصفوفات ذات البعدين A two-dimensional array سيكون لديها فهرسين,الاول يعرف(يحدد) الصفوف(ترتيب المدن في المصفوفة) والثاني سيعرف الأعمدة(درجة حرارة المدينة), لتتمكن من الوصول الى اسم درجة حرارة المدينة الثالثة في المصفوفة ثنائية البعد استخدم الفهارس التالية:

اسم المدينة الثالثة: Temperatures(2, 0)
متوسط درجة حرارة المدينة الثالثة: Temperatures(2, 1)



الشكل(5،2)يبين مصفوفتان-كل منهما أحادية البعد (على اليسار) والمصفوفة ثنائيةالبعد(ذات بعدين) المكافئة لهما(على اليمين وهي(درجات الحرارة(7،1))

الفائدة من استخدام المصفوفات المتعددة الأبعاد هو أنها وبشكل مبدئي اسهل في الادارة والترتيب,افترض انك تكتب لعبة ما. وتريد ان تضبط موقع قطع معينة على لوحة اللعب . كل مربع على اللوحة يتم تعينه بواسطة عددين :وهما الاحداثيات الافقية والعموديةhorizontal and vertical coordinates. التركيب الواضح لضبط مربعات اللوحة هو مصفوفة ثنائية البعد(مصفوفة ذات بعدين)two-dimensional array والتي فيها الفهرس الاول مطابق لعدد الصفوف, والفهرس الثاني مطابق لعدد الأعمدة.,

يمكن التصريح عن هكذا مصفوفة كما يلي: Dim Board(9, 9) As Integer فاللوحة مركبة من عشرة صفوف وعشرة أعمدة كماهو واضح في المصفوفة.

عندما يتم تحريك احد القطع من مربع في الصف الاول والعمود الاول الى المربع في الصف الثالث والعمود الخامس,تستطيع اسناد القيمة 0 للعنصر الذي يوافق الموقع الأولي(البدائي)(كان توشر هذا المربع برسم دائرة وبداخلها رقم 0 بواسطة قلم أحمر)

```
Board(0, 0) = 0
```

وتسند القيمة 1 الى المربع الذي تم تحريك القطعة اليه ليدل على الموقع الجديد من اللوحة☹(كأن تؤشر هذا المربع برسم دائرة وبداخلها الرقم واحد وذلك بواسطة قلم أحمر))

```
Board(2, 4) = 1
```

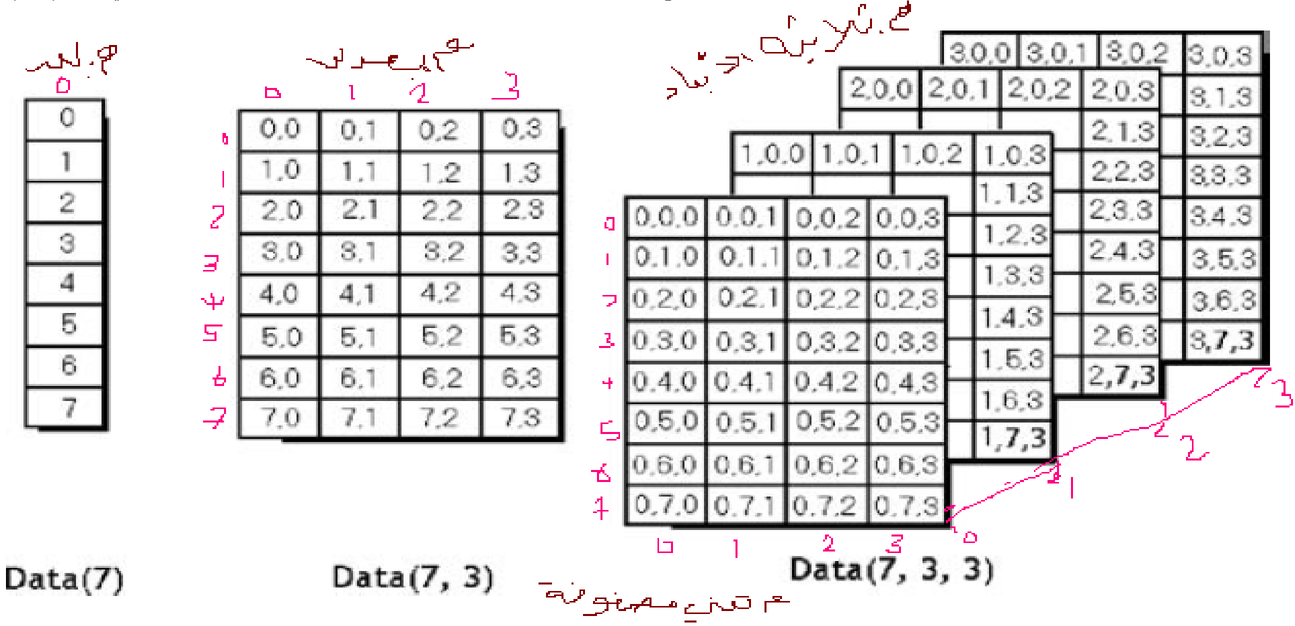
لتستكشف فيما اذا كانت القطعة على المربع اليساري العلوي top-left square عليك استخدام العبارة التالية(فالمكان الذي توجد فيه القطعة تسند له القيمة واحد والمكان الذي تغادره القطعة تسند له القيمة صفر(أو المكان الغير موجودة فيه القطعة)):

```
If Board(0, 0) = 1 Then
 { piece found}
    Else
 { empty square}
End If
```

هذه الملاحظة يمكن تطبيقها على باقي المصفوفات التي لها أكثر من بعدين فالعبارة التاليه تحمل مصفوفة ثلاثية لابعاد Matrix (القالب)(فيها 1000 عنصرفي كل بعد 10 عنصر أي ((10×10×10)

```
Dim Matrix(9, 9, 9)
```

يمكن ان تتخيل المصفوفة ثلاثية البعدthree-dimensional كمكعب مركب من صفائح مصفوفات ثنائية البعد overlaid two-dimensional كما يظهر في الشكل(6،2)



الشكل(6،2)رسم تخطيطي يبين (من اليسار الى اليمين)المصفوفات أحاديةالبعد-وثنائية البعد- وثلاثية البعد(موضحة مع الفهارس)

يمكن التمهيد للمصفوفة المتعددة الابعاد بعبارة واحدة, كما فعلت تماما مع المصفوفة احادية البعد,فقط عليك ادخال فواصل كافية في الاقواس التي تبع اسم المصفوفة لتدل على رتبة المصفوفة,فالعبارات التالية تمهد لمصفوفة ذات بعدين ومن ثم تطبع زرج من عناصرها:

```
Dim a(,) As Integer = {{10, 20, 30}, {11, 21, 31}, {12, 22, 32}}
      Console.WriteLine(a(0, 1)) ' سوف تطبع 20
      Console.WriteLine(a(2, 2)) ' سوف تطبع 32
```

ستعمل على تقسيم أسطر اسناد أبعاد المصفوفة في عدة اسطر لتجعل الكود أكثر قابلية للقراءة. فقط ادخل رمز تقسيم الاسطر(الشحطة المنخفضة)عند نهاية كل سطر متواصل كما يلي:

```
Dim a(,) As Integer = {{10, 20, 30}, _
                       {11, 21, 31}, _
                       {12, 22, 32}}
```

	0	1	2
0	(0,0)=10	(0,1)=20	(0,2)=30
1	(1,0)=11	(1,1)=21	(1,2)=31
2	(2,0)=12	(2,1)=22	(2,2)=32

الشكل(7،2)رسم توضيحي للمصفوفة a مع فهارس الخانات

إذا كان للمصفوفة اكثر من بعد, تستطيع استكشاف عدد الابعاد بواسطة الخاصية ()Rank,لنقول انك اعلنت عن مصفوفة لتخزين اسماء ورواتب المستخدمين باستخدام العبارة التالية:

```
Dim Employees(1,99) As Employee
```

لتعرف عدد أبعاد المصفوفة Employees استخدم العبارة التالية

```
Employees.Rank()
```

ولكن كي لايظهر خطأ, عليك اولاً تعريف تركيب واليك الكود كاملا :

```
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As _            System.EventArgs) Handles Button1.Click
      Dim Employees(1, 99) As Employee
      Debug.WriteLine(Employees.Rank.ToString())
     End Sub
   Structure employee
      Dim name As String
      Dim salary As Single
   End Structure
End Class
```

عندم تستخدم خاصية الطول Length property لاستكشاف عدد عناصر المصفوفة المتعددة الابعاد ستحصل على العدد الكلي للعناصر في المصفوفة(مثلا 2×100) ولكن لايجاد عدد العناصر في بعد معين, استخدم الطريقة GetLength method ومرر لها كمعامل نسبي البعد الذي تريد ايجاد عدد عناصره.. التعبير التالي سيعود بعدد العناصر في كل بعد من اتعاد المصفوفة:

`Debug.WriteLine(Employees.GetLength(0))` النتيجة ستكون في النافذة المباشرة للبعد الاول: 2

`Debug.WriteLine(Employees.GetLength(1))` النتيجة ستكون في النافذة المباشرة للبعد الثاني : 100

لان فهرس العنصر الاول وكما تعلمت مسبقا هو الصفر وفهرس العنصر الأخير هو طول المصفوفة مطروحا منه واحد,لنقول انك اعلنت عن مصفوفة بالعبارة التالية، لتخزين أرقام اللاعبين من اجل 15 لاعب , ويوجد خمس قيم(كل لاعب لديه خمس قمصان وكل قميص عليه رقم) لكل لاعب:

```
Dim Statistics(14, 4) As Integer
```

العبارات التالية ستعود بالقيم التالية المعلمة بالخط العريض في نافذة المخرجات:

```
Debug.WriteLine(Statistics.Rank)
2:    ' dimensions in array(ابعاد المصفوفة)
Debug.WriteLine(Statistics.Length)
75:    ' total elements in array (العدد الكلي لعناصر المصفوفة)
Debug.WriteLine(Statistics.GetLength(0))
15:    ' elements in first dimension (عدد العناصر في البعد الاول)
Debug.WriteLine(Statistics.GetLength(1))
5:    ' elements in second dimension (عدد العناصر في البعد الثاني)
Debug.WriteLine(Statistics.GetUpperBound(0))
14:    ' last index in the first dimension (الفهرس الاخير في البعد الاول)
Debug.WriteLine(Statistics.GetUpperBound(1))
4:    ' last index in the second dimension (الفهرس الاخير في البعد الثاني)
```

المصفوفات المتعددة الابعاد(أكثر من بعدين) قد بطل استخدامها obsolete لان مجمعات و مصفوفات لتراكيب محددة والكائنات(كما رايت عندما عرفنا مصفوفة من تركيب) اكثر مرونة وراحة(سواء كانت ذات بعدين أو بعد واحد)

المصفوفات الديناميكية Dynamic Arrays

في بعض الاحيان لا تعلم كم من العناصر ستضع في المصفوفة,فبدلا من جعل المصفوفة كبيرة بشكل كافي لحفظ عدد البيانات الاعظمي المتوقع (وهذا يعني وعلى الاغلب ان قسم من المصفوفة سيبقى فارغا), تستطيع التصريح عن مصفوفة ديناميكية(مرنة) dynamic array, وحجم المصفوفة الديناميكية يمكن ان يتغير خلال منهج البرنامج, .يمكن القول انك تحتاج لمصفوفة بعد ان يقوم المستخدم بادخال باقة البيانات التي يريد ,ومن ثم يكون قد عمل التطبيق على معالجتها وعرض نتائجها.

اذا فلماذا نحفظ كل البيانات في الذاكرة عندما لا يكون هناك حاجة لا ستخدامها؟ باستخدام لمصفوفات الديناميكية تستطيع ان تطرح البيانات وتعود بالمصادر التي كانت تشغلها الى النظام. لانشاء مصفوفة ديناميكية, صرح عنها وكالعادة بالعبارة Dim (أو عام Public أوPrivate)ولكن بقوسين فارغين وبدون ان تكتب طول المصفوفة:

```
Dim DynArray() As Integer
```

فيما بعد في البرنامج وعندما تعلم كم عدد العناصر التي تريد ان تخزنها في المصفوفة,استخدم العبارة ReDim لاعادة تحديد ابعاد المصفوفة,وهذه المرة ستكون بالحجم الحقيقي,.في المثال التالي: UserCount هي القيمة التي قام بادخالها المستخدم ويمكن ان تكون عدد صحيح ما (5,10,.....)

```
ReDim DynArray(UserCount)
```

العبارة ReDim يمكن ان تظهر فقط في اجراء ما. بشكل مختلف عن العبارة Dim,العبارة ReDim هي عبارة تنفيذية executable وتجبر التطبيق على تنفيذ فعل ما وقت التنفيذ.,

عبارات Dim ليست تنفيذية, ويمكن ان تظهر خارج نطاق الاجراءات.

ولا يمكن ايضا للمصفوفة الديناميكية ان يعاد تحديد ابعادها الى ابعاد متعددة, صرح عن مصفوفة ديناميكية بعبارة Dim خارج أي اجراء كما يلي :

```
Dim Matrix() As Double
```

ومن ثم فان استخدم العبارة ReDim في اجراء ما للتصريح عن مصفوفة ذات ثلاثة ابعاد هي عبارة خاطئة:

```
ReDim Matrix(9, 9, 9)
```
(لاحظ رسالة الخطأ التي يعرضها المترجم عندما تحرك المؤشر على هذه العبارة)

لاستخدام العبارة السابقة صرح عن مصفوفة ديناميكية ذات ثلاثة ابعاد فعبارة ReDim لا تسطيع تغير ابعاد المصفوفة كما يلي: Dim Matrix(,,) As Double أي يجب عليك ان تعلم عدد ابعاد المصفوفة الديناميكية ولو كنت لا تعلم سعة أو طول كل بعد(عددحدود كل بعد)

لاحظ ان العبارة ReDim لا تستطيع تغير نوع بيانات المصفوفة.ولذلك فان الشرط As غير موجود في عبارة ReDim.وأكثر من ذلك عبارات ReDim اللاحقة تستطيع تغير حدود المصفوفةMatrix ولكن ليس عدد ابعادها(مهمايكن لا يمكن تغيير عدد ابعاد المصفوفة). مثلا لا تستطيع ان تستخدم العبارة التالية(ReDim Matrix(99, 99 فيما بعد في كودك.

الكلمة المحجوزة حفظ أو حماية The Preserve Keyword

كل مرة تنفذ فيها عبارة ReDim, فان جميع القيم المخزنة حاليا فيها يتم فقدانها. و الفيجوال بيسك يعيد وضع قيم العناصر كما لو ان المصفوفة تم الاعلان عنها الآن(فيعيد اسناد العناصر العددية الى الصفر والعناصر النصية الى نص فارغ) ولكن تستطيع ومهما يكن ,ان تغير حجم المصفوفة بدون فقدان بياناتها باستخدام العبارةReDim مع الكلمة المحجوزة(حماية أو حفظ) Preserve التي تتميز بها العبارة ReDim والتي تجبر العبارة ReDim على اعادة تحجيم المصفوفة بدون طرح البيانات الموجودة مسبقا في المصفوفة. مثلا: تستطيع ان تزيد حجم المصفوفة(وليس عدد ابعادها) عنصر واحد بدون ان تفقد القيم الموجودة في العناصر كما يلي:

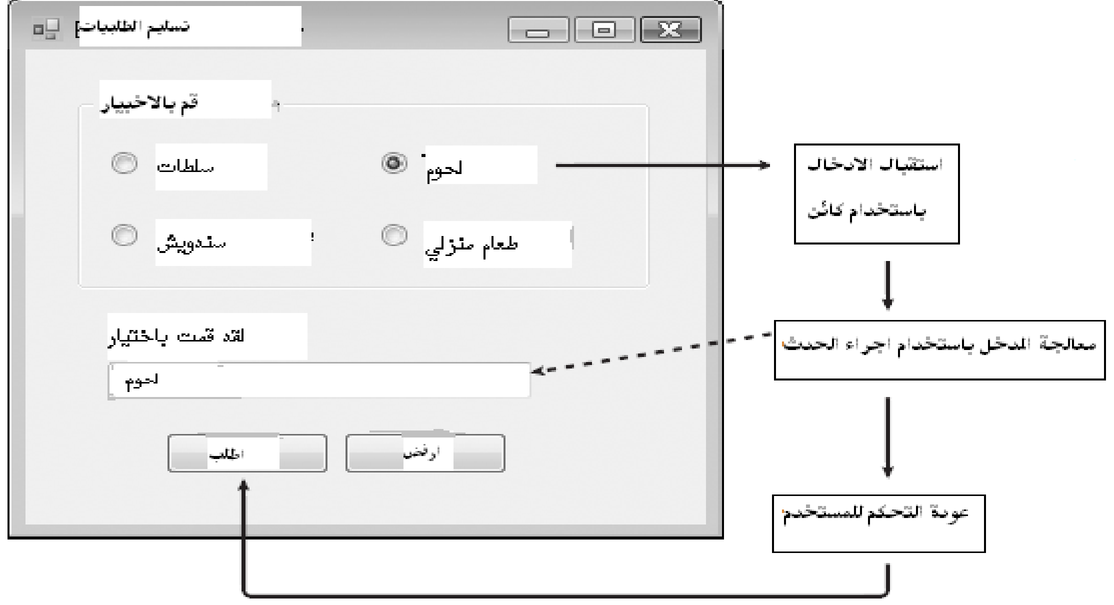
```
ReDim Preserve DynamicArray(DynArray.GetUpperBound(0) + 1)
```

فاذا كانت المصفوفة تحفظ 12 عنصر, فان العبارة السابقة ستضيف عنصرالى المصفوفة:وهو العنصر(DynamicArray(12 . بينماقيم العناصر ذات الفهارس من 0 الى 11 لن تتغير ابداً.

ملاحظة:سأطلق التسمية على مقطع من العبارات المحتواة ضمن إجراء ما أو ضمن جمل الشرط أو ضمن أي مقطع برمجي آخر(بلوك(block))وهي مفهومة برمجيا اكثر من أي مرادف أخر وهي بطبيعة الحال بلوك برمجي (مقطع برمجي لا يتجزأ)
كما اني أطلقت التسمية معامل نسبي على(argument)وارى انه المصطلح الأقرب برمجيا
بعد ان تتم هذا الفصل ستتكون قادر على ما فعل التالي:
-1 كتابة التعابير الشرطية
2 - استخدام جمل *If...Then* لتقسم البرنامج الى مجموعة من الجمل البرمجية بالاعتماد على شروط متنوعة.
3-استخدام اداة صندوق النص المقنع(المشفر) *MaskedTextBox* لاستقبال مدخلات المستخدم بتنسيق خاص.
4- استخدام جمل اختيار الحالة لاختيار خيار من عدة خيارات في كود البرنامج.
5- استخدام خاصية الاسم *Name* property لاعادة تسمية المشروع من ضمن البرنامج.
6- ادارة وترتيب حوادث الفارة وكتابة معالج حدث تنقلات الفارة *MouseHover* event handler .
في هذا الفصل ستتعلم كيف تقوم بعمل تفرع شرطي الى منطقة ما في برنامجك بالاعتماد على الدخل المستقبل من قبل المستخدم ,وسوف تتعلم ايضا كيف تقيّم واحدة او أكثر من الخاصيات او المتغيرات باستخدام تعابير الشرط.ومن ثم تنفذ واحدة او أكثر من جمل البرنامج بالاعتماد على النتيجة .باختصار ستزيد من مفرداتك البرمجية بإنشاء بلوكات(مقاطع موحدة) من الكود تسمى تراكيب القرار*decision structures* والتي تسيطر على كيفية تنفيذ برنامجك ,او مساره(سياقه بشكل داخلي(من خلال الكود))
الشيء الوحيد الذي يجب ان تكون قد تعلمته حتى الآن حول البرمجة في البيسك هو ان التطبيق مركب من مقاطع صغيرة محصورة,فالكود الذي تكتبه ليس إدراج لقائمة متراصة(موحدة) monolithic listing) بل هو مركب من مقاطع صغيرة والتي تدعى الإجراءات procedures وانك تعمل على إجراء واحد في كل مرة على حدة at a time .
نوعي الإجراءات المدعومة بواسطة البيسك هي موضوع بحثنا في هذا الفصل: الإجراءات الفرعية subroutines والوظائف functions .والتي هي قوام(بلوكات) بناء تطبيقاتك وسوف نناقش هذه الإجراءات بالتفصيل. من بين المواضيع الأخرى في هذا الفصل سوف تتعلم كيف تعمل ما يلي:
✓ استخدام عبارات التحكم بالسياق للفيجوال بيسك Use Visual Basic's flow-control statements
✓ كتابة الإجراءات الفرعية والوظائف Write subroutines and functions
✓ تمرير المعاملات النسبية للإجراءات الفرعية والوظائف Pass arguments to subroutines and functions

البرمجة المقادة بالأحداث Event-Driven Programming

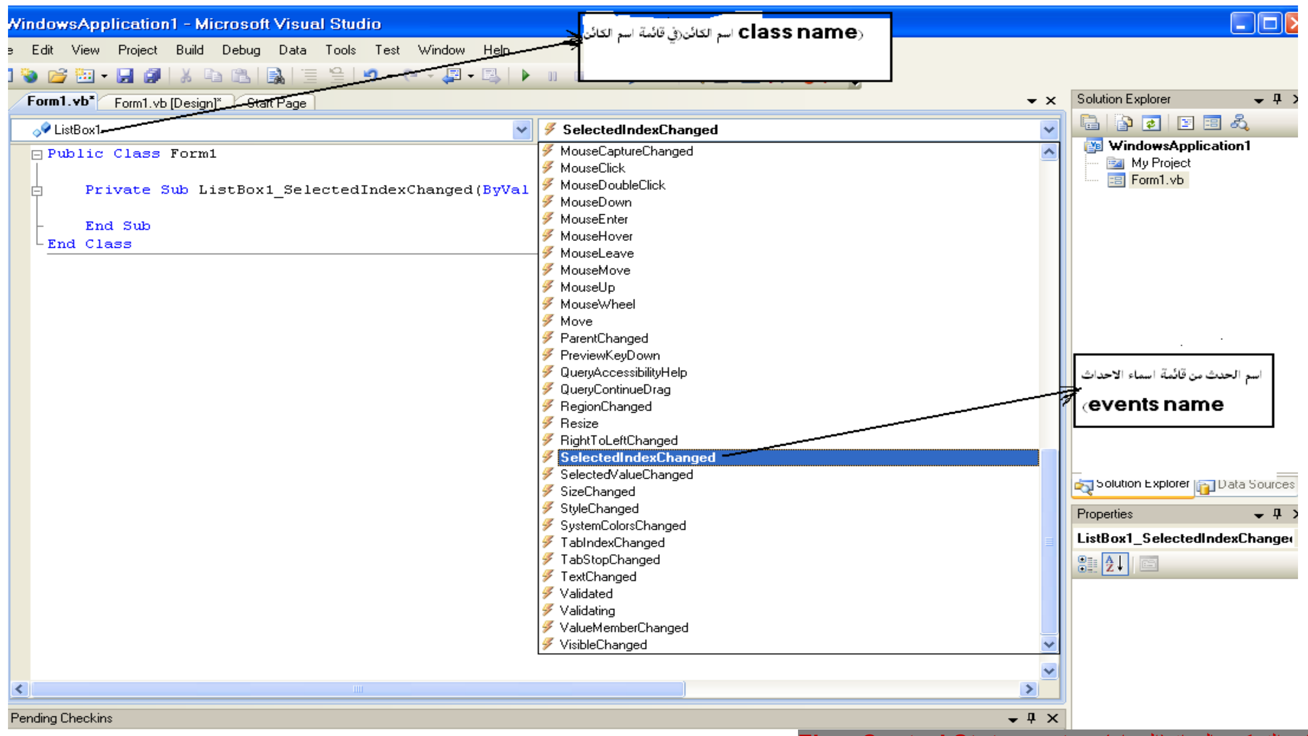
لقد عَرضَت البرامج التي كتبتها حتى الآن في هذا الكتاب أدوات صندوق النص Toolbox,القوائم menus,أشرطة الادوات toolbars, وصناديق الحوار dialog boxes , على الشاشة,وبهذه البرامج يستطيع المستخدم ان يعالج العناصر التي تظهر على الشاشة بأي ترتيب يراه مناسبا,ولكن تضع البرامج المستخدم تحت المسؤولية،عندما تنتظر بصبر إدخالات المستخدم (الإجابة),ومن ثم تعالج المدخلات بشكل تنبؤي predictably في الدارات البرمجية هذه الطرائق(الأساليب methodology)تعرف بالبرمجة المقادة بالأحداث event-driven programming , فأنت تعمل على بناء البرنامج بإنشاء مجموعة من الكائنات الذكية,والتي تعرف كيف تستجيب عندما يتفاعل معها المستخدم, ومن ثم يعمل البرنامج على معالجة مدخلات المستخدم باستخدام الإجراء المصاحب لهذه الكائنات,التخطيط التالي يريك كيف يعمل البرنامج المقاد بالحدث event-driven program في الفيجوال بيسك



يمكن ان تأتي إدخالات البرنامج من نظام الكمبيوتر نفسه,مثلا يمكن لبرنامجك ان يعلمك عندما يتم استقبال رسالة بريد الكتروني, او عندما تمضي فترة معينة على ساعة النظام,في هذه الحالة الكمبيوتر وليس المستخدم من يطلق هذه الأحداث, بغض النظر عن كيفية إطلاق الحدث , يستجيب الفيجوال بيسك باستدعاء الإجراء المصاحب للكائن الذي يميز هذا الحدث,ولقد تعاملت بشكل أولي مع الأحداث :النقر على الفارة Click,تغير الاختيار CheckedChanged, تغير الفهرس المختار SelectedIndexChanged مهما يكن, يستجيب الفيجوال بيسك الى العديد من أنواع الأحداث الأخرى
جوهر القيادة بالأحداث في الفيجوال بيسك يعني ان معظم عمليات الحوسبة التي يتم عملها في برامجك يتم انجازها بواسطة إجراءات الحدث., هذه القطع(بلوكات) من الكود المخصص للحدث تعالج المدخلات, تحسب القيم الجديدة, تعرض الاخراجات, وتعالج العديد من المهام الأخرى.
ستتعلم في هذا الفصل, كيف تستخدم تراكيب القرار لتقارن (تضاهي) المتغيرات,او الخصائص,والقيم,.وكيف تنفذ واحد او أكثر من العبارات بالاعتماد على النتائج ,ومن ثم ستستخدم الحلقات والمؤقتات,فستستخدم الحلقات لتنفيذ مجموعة من الجمل مرة بعد الأخرى حتى يتحقق الشرط او يصبح شرط ما صحيح ,.بوجود تراكيب فعالة جدا مثل تراكيب التحكم والتي تساعدك على بناء إجراءات الحدث الخاصة بك, والتي تمنح هذه الإجراءات الإمكانية للاستجابة لأي موقف مهما كان.

الأحداث المدعومة بواسطة كائنات الفيجوال بيسك: Events Supported by Visual Basic Objects

كل كائن ي الفيجوال بيسك لديه مجموعة معرّفة مسبقا من الأحداث والتي يستطيع ان يستجيب لها,وهذه الأحداث يتم جدولتها عندما تختار اسم الكائن من قائمة اسم الكائن في أعلى محرر الكود ومن ثم تختار سهم اسم الطريقة (الأحداث تكون محددة بشكل مرئي في الفيجوال استوديو بواسطة الأيقونة التي تكون على شكل برغي مضيء)كما يوضحه الشكل المرافق ,تستطيع ان تكتب إجراء الحدث لأي من هذه الأحداث,وإذا ما حدث ذلك الحدث في البرنامج,فان الفيجوال بيسك سينفذ هذا الإجراء المصاحب له,مثلا كائن صندوق القائمة listbox يدعم أكثر من 60 حدث ومن ضمنها'النقر Click, النقر المزدوجDoubleClick, سحب وإسقاط DragDrop, سحب فوق DragOver, التركيزGotFocus , النقر من خلال لوحة المفاتيح من خلال مفاتيح مخصصة KeyDown,ضغط مفتاح من المفاتيحKeyPress, الضغط بمفاتيح محدد مع الرفع KeyUp,مغادرة الأداة LostFocus,تخصيص زر الفارةMouseDown, حركة الفارة MouseMove, تخصيص زر الفارة مع مفتاح معين MouseUp, تنقلات الفارة MouseHover,تغير النص TextChanged, والتحققValidated من المكن ألا تحتاج لكتابة الكود لجميع هذه الاحداث, من الممكن ان تحتاج كتابة كود الى اثنين او ثلاثة منها وحتى أربعة فقط في تطبيقاتك, ولكن من الجيد ان تعلم ان لديك الكثير من الخيارات,عندما تقوم بإنشاء كائن ما(كل شيء كما قلنا سابقا في الفيجوال بيسك يعتبر كائنا وهذه هي البرمجة كائنية التوجه object- oriented programming فالأداة على واجهة المستخدم هي كائن له خصائصه وطرقه التي تميزه) على الفورم.فالشكل التالي يوضح الكلام السابق فهو يظهر قائمة جزئية بالأحداث لكائن(اداة)صندوق القائمة في محرر الكود:



عبارات التحكم بالسياق(المسار) Flow-Control Statements

ما يجعل لغات البرمجة مرنة تماما وقادرة على التعامل مع تحديات البرمجة و كل موقف(حالة) مع وجود مجموعة صغيرة نسبيا من الأوامر هو قدرتها على اختبار الشروط الخارجية أو الداخلية والتصرف تبعا لذلك,. البرامج ليست مجموعات موحدة monolithic sets من الأوامر التي تُجري carry out نفس الحسابات كل مرة يتم تنفيذها executed ، هذا ما تقوم به الآلات الحاسبة(وبشكل مفرط البرامج البسيطة), بالمقابل إنها تقوم بتبديل(ضبط) سلوكها بالاعتماد على البيانات المزودة بها, في حالة الشروط الخارجية مثل النقر بالفارة أو بوجود جهاز طرفي peripheral ، حتى في الظروف غير العادية الناتجة عن البرنامج نفسه.

في الواقع ,العبارات التي تم مناقشتها في النصف الأول من هذا الفصل هي على ماذا تقوم البرمجة. وبدون المقدرة على السيطرة على سياق(مسار) البرنامج فان الكمبيوترات مجرد آلات حاسبة كبيرة يصعب تحريكها فقط., ولقد رأيت كيف تستخدم العبارة if لتحول مسار التنفيذ في الفصول السابقة.في هذا الفصل ستجد مناقشة كاملة لعبارات التحكم بالسياق أو المسار هذه العبارات تم تجميعها في تصنيفين رئيسين وهما:عبارات الفصل(الحكم) Decision Statements وعبارات التكرار(الدوران)Looping statements

جمل الفصل (القرار) Decision Statements

تحتاج التطبيقات الى آلية لاختبار الشروط, واخذ مسار مختلف لفعل ما بالاعتماد على حصيلة(نتيجة الاختبار).تزود الفيجوال بيسك بثلاثة من جمل الحكم decision أو جمل الشرط conditional وهي:

- If. . .Then (اذا.........عندئذ)
- If. . .Then. . .Else(اذا.....عندئذ.....وإلا)
- Select Case (اختر حالة)

واحد من أكثر الادوات فائدة في معالجة المعلومات في إجراء الحدث هو التعبير الشرطي ,التعبير الشرطي هو جزء من كود البرنامج الكامل والذي يسال بسؤال True-or-False بصح او خطأ فيما يخص خاصية ما, او متغير او أي جزء أخر من كود البرنامج,مثلا, التعبير الشرطي Price < 100يتم تقيمه الى صح اذا كان سعر Price المتغير الذي يحوي قيمة ما اقل من 100 ويتم تقيمه الى خطأ اذا كان السعر Price يحتوي على قيمة اكبر من او تساوي 100 . تستطيع استخدام معاملات المضاهاة التالية في التعبير الشرطي:

Comparison operator (معامل المقارنة)		Meaning (المعنى)
=	يساوي الى	Equal to
<>	لا يساوي الى	Not equal to
>	(من اليسار الى اليمين) اكبر من	Greater than
<	(من اليسار الى اليمين) اقل من	Less than
>=	(من اليسار الى اليمين) اكبر من او يساوي	Greater than or equal to
<=	(من اليسار الى اليمين) اقل من او يساوي الى	Less than or equal to

الجدول التالي يريك بعض تعابير الشرط ونتائجها وسوف تعمل العديد من تعابير الشرط في هذا الفصل حتى تفهمها جيداً"

Conditional expression (تعبير الشرط)	Result (النتيجة)
`10 <> 20`	True (10 is not equal to 20) صح لان 10 لا يساوي 20
`Score < 20`	True if *Score* is less than 20; otherwise, False صح اذا كانت المحصلة اقل من 20 وغير ذلك فهي خطأ
`Score = Label1.Text`	True if the *Text* property of the *Label1* object contains the same value as the *Score* variable; otherwise, False صح اذا كانت خاصية النص لكائن العنوان تحوي على نفس القيمة ،والا فهي خطأ
`TextBox1.Text = "Bill"`	True if the word “Bill” is in the *TextBox1* object; otherwise, False صح اذا كانت الكلمة "بيل" في خاصية النص لكائن صندوق النص والا فهي خطأ

الشرط (اذا.........عندئذ) **If...Then**

عندما يُستخدم تعبير الشرط في بلوك من الجمل يدعى تركيب القرار *decision structure*, انه يتحكم فيما اذا تم تنفيذ عبارات أخرى في برنامجك و بأي ترتيب تم تنفيذها, تستطيع استخدام التركيب *If...Then* لتقيم شرط في برنامج و تأخذ سياق (منهج) الفعل بالاعتماد على النتيجة. ابسط شكل للتركيب *If...Then* يتم كتابته في سطر واحد:

اذا شرط عندئذ If condition Then statement

حيث ان شرط *condition*تعبير شرطي, والجملة *statement*هي قيمة عبارة في برنامج الفيجوال بيسك,مثلا: If Score >= 20 Then Label1.Text = "You win!"

كما في التركيب *If...Then*الذي يستخدم تعبير الشرط الذي طالعناه سابقا Score >= 20

لتحدد فيم اذا البرنامج سيسند خاصية النص *Text* لكائن العنوان *Label1* الى "أنت ربحت" "You win!" فإذا كان المتغير المحصلة *Score* يحتوي على قيمة اكبر من او تساوي الى 20 فان فيجوال بيسك يسند خاصية النص وإلا فانه يتجاوز إسناد الجملة وينفذ السطر التالي في الإجراء,هذا النوع من المقارنة (المضاهاة) دائما ينتج عنها قيمة صح او خطأ, فتعبير الشرط conditional expression لا ينتج عنه يمكن(من المحتمل maybe) أبدا.

العبارة If. . .Then تختبر تعبير ما,والذي يُعرف كشرط condition ،فإذا كان الشرط صحيح فان البرنامج يعمل على تنفيذ العبارة أو العبارات التي تتبع الشرط ،يمكن لعبارة If. . Then. ان يكون لها تركيب سطر مفرد أو متعدد الأسطر .

الشروط تعبير منطقي logical expressions يتم تقيمه الى إحدى القيمتين فقط(صح/خطأ) وهي بالعادة تحتوي على معاملات مقارنة comparison operators مثل[(المساواة (=)equals],[الاختلاف(عدم التساوي)(<>) different)] و[(اقل من)(<) less than)] و[وأكبر من (>) greater than] و[أقل من او يساوي(=>) equal to [less than وهكذا ومعاملات منطقية مثل [و And] و[أو Or] و[أو الاختبار Xor] و[ليس Not] إليك بعض الأمثلة لشروط محققة (صحيحة)☹ (لا تنسى ان تصرح عن المتغيرات قبل استخدامها كما تعلمت سابقا فالمتغير age1 صرح عنه كعدد صحيح وexpDate صرح عنه كتاريخ وهكذا في بعض الاحيان استخدم بعض المتغيرات كأمثلة ومن البديهي اذا ما أردت ان تجربها ان تصرح عنها)

```
If (age1 < age2) And (age1 > 12) Then ...
```

```
If score1 = score2 Then ...
```

الأقواس ليست ضرورية في المثال الأول,ولكن لجعل الكود أسهل قليلا بالنسبة للقراءة.. في بعض الأحيان تكون الأقواس إلزامية(إجبارية mandatory) لتحديد الترتيب الذي يتم به تقيم أجزاء التعبير, تماما بشكل مشابه للصيغ الرياضية والتي تتطلب أقواس لتدل على الأولوية في الحسابات,.تستطيع ايضا تنفيذ جمل متعددة وذلك بالفصل بينها بالنقطتان العموديتان colons

```
If condition Then statement: statement: statement
```

اليك مثال عن عبارة if مفردة السطر:

```
expDate = expDate + 1
If expdate.Month > 12 Then expYear = expYear + 1 : expMonth = 1
```

تستطيع ان تجزأ هذه العبارة في عدة سطور باستخدام التركيب المتعدد السطور والذي يحدد حدود العبارات التي يجب تنفيذها كشرط بالعبارة End If كما هو مبين هنا:

```
If expDate.Month > 12 Then
 expYear = expYear + 1
 expMonth = 1
End If
```

خاصية الشهرMonth property لنوع بيانات التاريخ Date type تعود بشهر التاريخ الذي تم تطبيقها عليه كـــقيمة رقمية. معظم مبرمجي الفيجوال بيسك يفضلون التركيب المتعدد السطور لعبارة If statement حتى ولو كانت تحتوي على جملة وحيدة(شرط واحد فقط). قوام الجمل بين الكلمات المحجوزة Then و End If يشكل جسم الجملة الشرطية ويمكن ان يكون لديك العديد من العبارات في الجسم اذا احتجت لها.

ملاحظة:

في حالة التعامل مع متغيّر منطقيّ، يمكن استخدام جملة الشرط كالتالي:

```
Dim Check As Boolean = True
If Check = True Then Check = False
If Check = False Then Exit Sub
```

لاختصار الكتابة، يعتبر VB أن قيمة المتغيّر المنطقيّ = True إذا لم يذكر ذلك صراحة.. بمعنى أنّ الجملة التالية:

```
If Check Then Check = False
```

مكافئة تماما للجملة التالية:

```
If Check = True Then Check = False
```

العديد من خاصيات الأدوات هي قيم منطقية والتي يتم تقيمها الى قيمة (صح/خطأ True/False),لنقول ان الواجهة لديك تحتوي على أداة صندوق اختيار CheckBox و أردت ان تضع عنوان لها (فعال/معطل On or Off)وذلك للدلالة على انها مختارة حاليا أو لا.إليك جملة الشرط التي تغير عنوان اداة صندوق الاختيار:

```
If CheckBox1.Checked Then
        CheckBox1.Text = "ON"
     Else
        CheckBox1.Text = "OFF"
End If
```

هذه العبارة تغير عنوان أداة صندوق الاختيار بشكل جيد , ولكن متى سيتم تنفيذها؟ ادخل العبارة في معالج حدث(تغير الاختيار CheckedChanged) لاداة صندوق الاختيار والذي يتم إطلاقه في كل مرة علامة الاختبار(الصح)للأداة يتم تبديلها الى(تفعيل/تعطيل on or off او علامة الصح موجودة/غير موجودة)إما بسبب عمل المستخدم على الوجهة أو من ضمن الكود. يمكن للتعابير ان تصبح اكثر تعقيدا,فالتعبير التالي يتم تقيمه الى صح اذا كان المتغير *date1* يمثل تاريخ قبل التاريخ 2008 وفيماذا ايضا كان واحد من المتغيران(مجموع النقاط المحرزة) *score1*و *score2* لا يتجاوز الـ90 :

```
If (date1 < #1/1/2008) And (score1 < 90 Or score2 < 90) Then
        ' statements
End If
```

الأقواس حول القسم الأخير من المقارنة إجباري,لأننا نريد من المترجم ان ينجز المقارنة التالية اولا:

```
score1 < 90 Or score2 < 90
```

فأيا من المتغيرين يقل عن الـ90 فان التعبير السابق يقيم الى صح(True) والشرط الأولي يتم تخفيضه الى التالي:

```
If (date1 < #1/1/2008) And (True) Then
```

المترجم سيعمل على تقييم الجزء الأول من التعبير(والذي يضاهي تاريخين)وأخيرا سيعمل على دمج قيمتين منطقيتين بواسطة المعامل (And) operator:فإذا كانت كلا القيمتين True صح فان التعبير الكلي صح وإلا فهو خاطئ False . اذا لم تستخدم الأقواس فان المترجم سوف يقيم الأجزاء الثلاثة من التعبير كمايلي:

التعبير الأول:date1 < #1/1/2008#

التعبير الثاني:score1 < 90

التعبير الثالث:score2 < 90

ومن ثم فانه سيدمج التعبير الأول مع التعبير الثاني باستخدام المعامل And وأخيرا سوف يجمع النتيجة مع المعامل الثالث باستخدام المعامل OR . فإذا كان *score2* اقل من 90 فان التعبير بالكامل سيتم تقيمه الى صح True بغض النظر عن القيمة في متغير التاريخ date1 variable

الشرط (اذا.....عندإذٍ.....وإلا) If. . .Then. . .Else

تدعم الفيجوال بيسك تركيب القرار الذي تستطيع استخدامه لتضمّن عدة تعابير شرط,وهذا المقطع من الجمل يمكن ان يحوي عدة سطور طويلة وتحتوي على كلمات محجوزة هامة مثل وإلا *Else* او(و الااذا *ElseIf*)ونهاية الشرط *End If* والتركيب العام التالي يوضح ذلك:

```
If condition1 Then
statements executed if condition1 is True العبارات التي ستنفذ اذا تحقق الشرط الأول
        ElseIf condition2 Then
statements executed if condition2 is True العبارات التي ستنفذ اذا تحقق الشرط الثاني
[Additional ElseIf conditions and statements can be placed here] يمكن إضافة عبارات
إضافية (والااذا) هنا
        Else
statements executed if none of the conditions is True
End If
```

في هذا التركيب الشرط الأول condition1 يتم تقيمه أولا,فإذا كان تعبير الشرط هذا صحيحا,فان مقطع الجمل التي تحته سيتم تنفيذها عبارة عبارة ويتجاوز البرنامج باقي العبارات ومن ثم ينتقل الى تنفيذ الكود الذي بعد الكلمة المحجوزة End If (تستطيع ان تضمن واحد او اكثر من عبارات البرنامج) .أما اذا كان الشرط الأول غير صحيح فان الشرط الثاني(condition2)سيتم تقيمه,فإذا كان صحيحا,فان البلوك الثاني من الجمل سيتم تنفيذه(تستطيع إضافة شروط ElseIf كما تحب او تقضيه الحاجة لذلك اذا كان لديك شروط أخرى تحتاج الى تقيم).أخيرا اذا لم يكن ولا واحد من هذه الشروط صحيحا,فان الجمل تحت الكلمة المحجوزة Else سيتم تنفيذها,وبعدها سيتم إغلاق كل التركيب بعبارة الكلمة المحجوزة End If.

الكود التالي يريك تركيب الشرط متعدد الأسطر والذي يمكن ان يستخدم لتحديد كمية الضرائب تبعا لعائد الضرائب المقدم (قائم على افتراض hypothetical)(أرقام الدخل والنسبة المئوية هي من ملحق متوسط الضرائب لمشروع خدمة عائدات الدخل في الولايات المتحدة 2007 projected United States Internal Revenue Service 2007 Tax Rate Schedule لإملاء ضريبي مفرد) حيث: AdjustedIncome :متغير ضبط الدخل,TaxDue :متغير الضريبة المقطوعة

```
Dim AdjustedIncome, TaxDue As Double
        AdjustedIncome = 50000
If AdjustedIncome <= 7825 Then '10% tax bracket(%10 حسم بنسبة)
            TaxDue = AdjustedIncome * 0.1
        ElseIf AdjustedIncome <= 31850 Then '15% tax bracket(%15 حسم بنسبة)
            TaxDue = 782.5 + ((AdjustedIncome - 7825) * 0.15)
        ElseIf AdjustedIncome <= 77100 Then '25% tax bracket(%25 حسم بنسبة)
            TaxDue = 4386.25 + ((AdjustedIncome - 31850) * 0.25)
        ElseIf AdjustedIncome <= 160850 Then '28% tax bracket(%28 حسم بنسبة)
            TaxDue = 15698.75 + ((AdjustedIncome - 77100) * 0.28)
        ElseIf AdjustedIncome <= 349700 Then '33% tax bracket(%33 حسم بنسبة)
            TaxDue = 39148.75 + ((AdjustedIncome - 160850) * 0.33)
        Else '35% tax bracket                            (%35 حسم بنسبة)
            TaxDue = 101469.25 + ((AdjustedIncome - 349700) * 0.35)
End If
```

ملاحظة هامة : ترتيب تعابير الشرط في التركيب *If...Then* والجمل *ElseIf* خطير ويستدعي الانتباه, فماذا يحدث اذا ما عكست ترتيب عبارات الشرط في مثال حساب الضريبة, وعملت على جدولت النسب في التركيب من الأعلى الى الأدنى؟ فدافعي الضرائب في الضرائب المقطوعة بالنسبة 10 و النسبة 15والنسبة 25والنسبة28 و النسبة 33 . جميعها موضوعة في نسبة حسم الضريبة الأخير 35 لان جميعها لديها العائد اقل او يساوي الى$349,700(يتوقف الفيجوال بيسك عند الشرط الأول اذا كان صحيحا, حتى ولو كانت الشروط الأخرى صحيحة,)لان التعابير الشرطية في هذا المثال تختبر نفس المتغير فهي تحتاج الى جدولة بترتيب تصاعدي ascending(من الأدنى الى الأعلى,) لوضع دافعي الضرائب تماما في المكان المناسب,. معنى الكلام,عندما تستخدم تعبير شرطي واحد انتبه الى الترتيب بحذر.فإذا صادف البرنامج في مثل هذه التعابير ان تحقق الشرط في احد العبارات الاولى فان فيجوال بيسك لن يكمل التحقق من العبارات الأخرى بل ينتقل الى تنفيذ العبارات والجمل التي تلي نهاية الشرط End If مباشرة

تركيب القرار هذا يختبر المتغير الذي من نوع الدقة المزدوجة double-precision variable وهو المتغير AdjustedIncome من المستوى الأول للدخل والمستويات اللاحقة حتى يتحقق واحد من تعابير الشرط(أي يصادف ان يكون صح) ومن ثم يحدد ضريبة الدخل التي على دافع الضريبة ان يدفعها تبعا لذلك,. مع بعض التعديلات البسيطة, يمكن لهذا الكود ان يستخدم لحساب الضريبة المستحقة owed على أي دافع ضرائب taxpayer في نظام ضريبي متقدم مثل هذا المستخدم في الولايات المتحدة مع تزويد نسب الضرائب الكاملة وتحديثها وبالتالي فان القيمة في المتغير AdjustedIncome يتم تصحيحها,. البرنامج السابق المكتوب سيعطي الضريبة المستحقة الصحيحة من اجل كل دافع ضرائب في الولايات المتحدة لعام2007, فإذا ما تم تغير نسبة الضريبة(نسبة الحسم) فالأمر البسيط هو تحديث عبارات الشرط بتراكيب شرط إضافية لتحديد الإملاء الضريبي لدافعي الضرائب taxpayers' filing status. يمكن للبرنامج ان يوسع نفسه بسهولة readily ليحتوي جميع دافعي الضرائب في الولايات المتحدة.

ملاحظة: التعابير التي يتم تقيمها الى صح او خطأ تسمى ايضا بالتعابير المنطقية *Boolean expressions*, والنتيجة صح او خطأ يمكن ان يتم إسنادها الى متغير من النوع المنطقي او الى خاصية,فتستطيع ان تسند القيم المنطقية Boolean values الى خاصية كائن معينobject properties, او الى متغيرات منطقية Boolean variables والتي تم انشاءها باستخدام عبارة *Dim*والكلمة المحجوزة *As Boolean*.

تدريب

التحقق من المستخدمين باستخدام التركيب Validate users by using *If...Then*

1-شغل الفيجوال استوديو وانشأ مشروع تطبيق نماذج ويندوز جديد وسمه (التحقق من مستخدمي),تلاحظ انه تم إنشاء المشروع الجديد و blank form لوحة النموذج الفارغة تفتح في المصمم(Designer)

2-اضغط على الفورم وضع في خاصية النص *Text* للفورم form (التحقق من المستخدم)

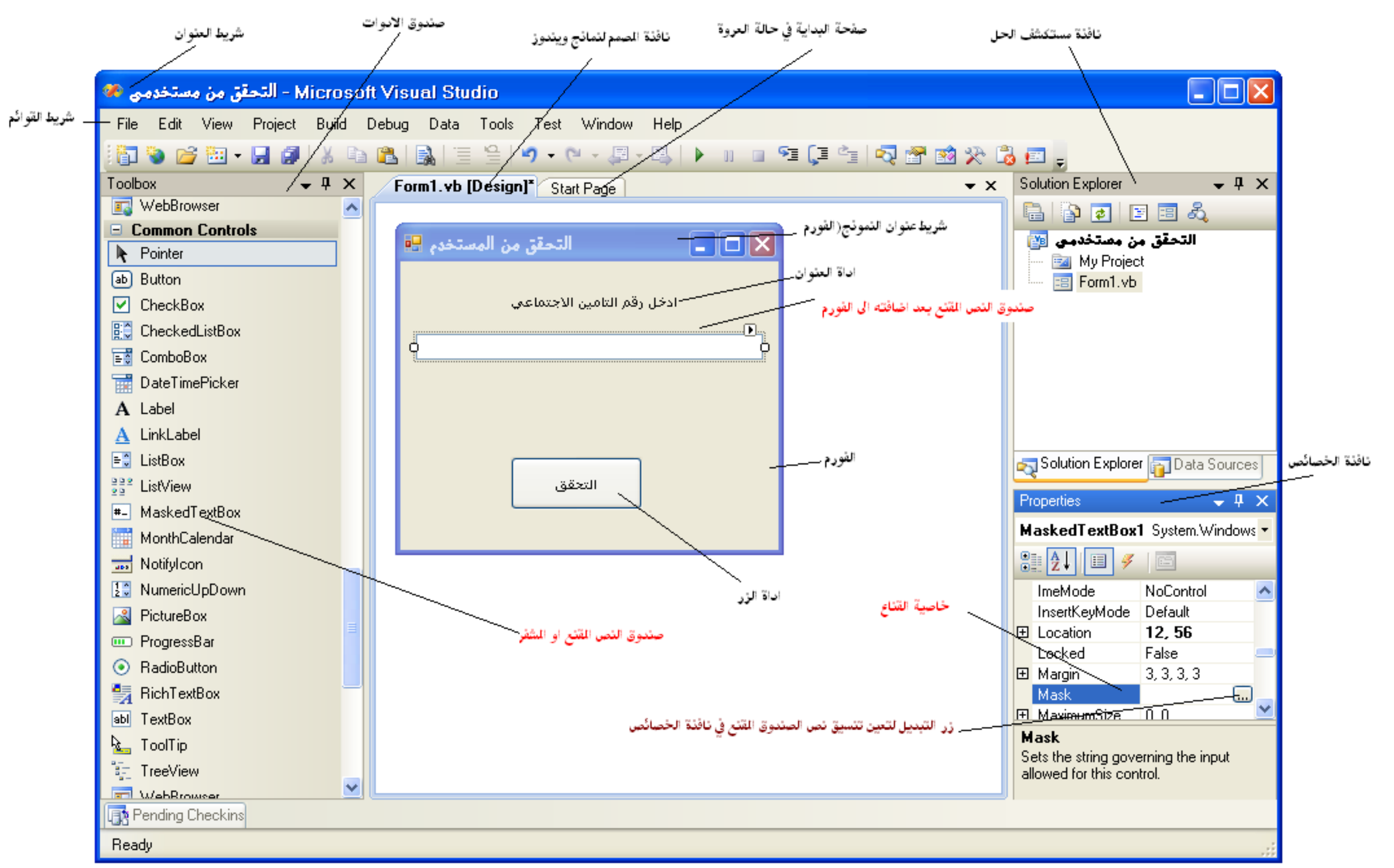
3- استخدم اداة العنوان Label من صندوق الادوات toolbox (اسحب هذه الأداة الى الفورم (سحب وترك) لإنشاء عنوان (label على الفورم form, واستخدم نافذة الخصائص Properties window لوضع خاصية Textالنص لهذه الأداة الى ("ادخل رقم التامين الاجتماعي")

4- كرر الخطوة السابقة لإضافة اداة الزر *Button*الى الفورم وضع النص التالي له في خاصية *Text* النص " التحقق "

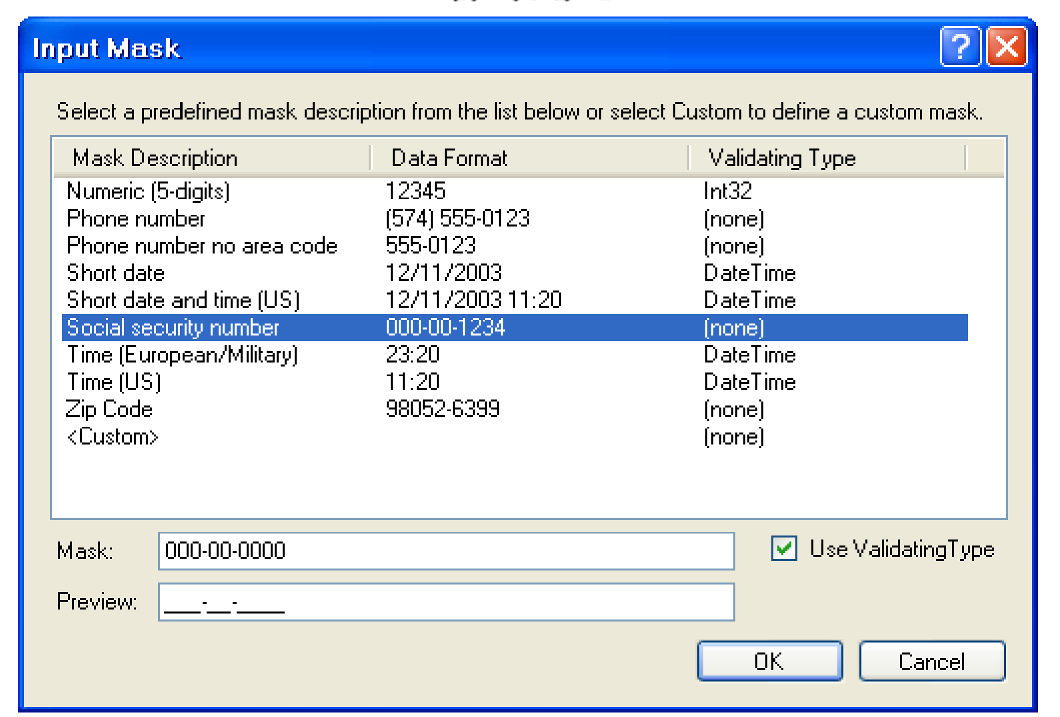
5- كرر الخطوة السابقة لإضافة اداة صندوق نص مشفر *MaskedTextBox* من صندوق الادوات Toolbox تحت عروة الادوات العامةCommon Controls tab , وضعه تحت اداة العنوانlabel :حيث ان صندوق النص المشفر مشابه تماما للأداة صندوق النص والتي استخدمتها من قبل,ولكن باستخدام صندوق النص المشفر تستطيع التحكم بتنسيق النص وذلك بإعداد خاصية القناع *Mask* وتستطيع استخدام تنسيقات موجودة مسبقا التي تزودك بها الأداة نفسها او تختار تنسيقك الخاص.وسوف تستخدم هذه الأداة في البرنامج لتطلب من المستخدمين إدخال رقم التامين الاجتماعي في تنسيق 9 أرقام قياسية المستخدمة في خدمة الدخل القومي للولايات المتحدة.

6- في الكائن صندوق النص المشفر 1 MaskedTextBox1 الذي تم إضافته الى الفورم اضغط على الخاصية قناع Maskمن نافذة الخصائص واضغط زر التبديل ellipses المجاور لها سيفتح صندوق حوار إدخال القناع Input Mask مظهرا لك قائمة بقوالب التنسيق المعرفة مسبقا او الأقنعة masks. انقر على رقم التامين الاجتماعي في القائمة Social Security Number

ما عملته حتى الآن موضح في الشكل التالي وصندوق حوار إدخال القناع موضح في الشكل الذي يليه:



بيئة تطوير الفيجوال استوديو



على الرغم من أني لن استخدمه الآن ولكن خذ دقيقة ولاحظ الخيار<Custom> والذي تستطيع استخدامه فيما بعد لإنشاء أقنعة خاصة بك باستخدام الأعداد ورموز placeholder حافظة المكان مثل الواصلة(-) hyphen

7- اضغط لقبول رقم التامين الاجتماعي كـقناع إدخال لك,. تعرض الفيجوال استوديو قناع الإدخال الذي اخترته في الكائن *MaskedTextBox1* كما يظهر في الشكل التالي:



تنسيق القناع الذي اخترته يظهر في اداة صندوق النص المقنع

8- اضغط مزدوج على زر التحقق لتفتح عروة نافذة محرر الكود والذي ينقلك مباشرة الى معالج حدث ضغط الزر:

عروة نافذة محرر الكود

معالج حدث ضغط الزر حيث ستكتب كودك ضمنه

التحقق من مستخدمي - Microsoft Visual Studio

File Edit View Project Build Debug Data Tools Test Window Help

Toolbox | General

There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.

Start Page | Form1.vb | Form1.vb [Design]

Button1 | Click

```
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As Syster
        If MaskedTextBox1.Text = "555-55-1212" Then
            MsgBox("يا مرحبا بك!")
        Else
            MsgBox(" لا استطيع تمييز الرقم")
        End If

    End Sub
End Class
```

Pending Checkins

Ready | Ln 8 | Col 15 | Ch 15 | INS

9- اكتب الكود التالي في هذا الحدث:

```
If MaskedTextBox1.Text = "555-55-1212" Then
            MsgBox("يا مرحبا بك!")
        Else
            MsgBox(" لا استطيع تمييز الرقم")
End If
```

هذا التركيب البسيط للقرار *If...Then* يختبر قيمة خاصية النص *Text* لكائن فإذا كانت مساوية "555-55-1212" فان التركيب سيعرض رسالة ترحب بك("يا مرحبا بك ! ") أما اذا كان الرقم المدخل بواسطة المستخدم قيمة أخرى سيعرض التركيب رسالة "لا استطيع تمييز الرقم", ما يميز هذا البرنامج بالرغم من بساطته هو كيف يعمل الكائن على فلترت إدخالات المستخدمين بشكل ألي الى التنسيق الصحيح فلا يحتاج المستخدم الى وضع فواصل او أي رموز أخرى يعمل هذا الكائن على وضع هذه التنسيقات.

10- اضغط الأمر Save All حفظ الجميع على شريط الادوات القياسية Standard toolbar لحفظ المشروع والتغيرات التي قمت بها, حدد المكان الذي تريد حفظ المشروع فيه..

11- اضغط الزر ابدأ التصحيح Start Debugging من على شريط الادوات القياسية Standard toolbar او بشكل اختياري اضغط F5 ليتم تشغيل البرنامج في بيئة التطوير المتكاملةIDE , تطلب الفوم من المستخدم إدخال رقم التامين الاجتماعي SSN في التنسيق المناسب,وتعرض الخطوط المنخفضة والوصلات underlines and hyphens لتقدم للمستخدم تلميح عن التنسيق المطلوب.

12- اكتب مثلا " ا ب ت " لتختبر قناع الإدخال ماذا ترى يعمل فيجوال بيسك على منع كتابة الأحرف لان الأحرف لا تناسب التنسيق المطلوب حيث ان رقم التامين الاجتماعي ذو التسع أرقام هو المطلوب.

13- جرب البرنامج بأرقام متنوعة لترى كيف يعمل البرنامج ومن ثم اعمل على إغلاقه.

Using Logical Operators in Conditional Expressions:استخدام المعاملات المنطقية في التعابير الشرطية

تستطيع اختبار أكثر من تعبير شرط واحد في العبارات If...Then و ElseIf اذا كنت تريد ان تضمن أكثر من معيار اختيار selection criterion في تركيب القرار decision structure, فان الشروط الإضافية يتم ربطها مع بعضها البعض باستخدام واحد او أكثر من المعاملات المنطقية المجدولة في الجدول التالي :

Logical operator المعامل المنطقي	Meaning المعنى
And اذا كان كلا الشرطان صح ستكون النتيجة صح	both conditional expressions are True, then the result is True.
Or اذا كان اي من الشرطان صحيح ستكون النتيجة صح	If either conditional expression is True, then the result is True.
Not اذا كان الشرط خطأ ستكون النتيجة صح اما اذا كان الشرط صح ستكون النتيجة خطأ	If the conditional expression is False, then the result is True. If the conditional expression is True, then the result is False.
Xor	If one and only one of the conditional expressions is True, then the result is True. If both are True or both are False, then the result is False. (*Xor* stands for exclusive *Or*.) اذا كان واحد وفقط واحد من الشروط صح فستكون النتيجة صح؛اما اذا كان كلا الشرطان صح او كلاهما خطأ عندها ستكون النتيجة خطأ؛(وهو بديل مكان او الخاص)

ملاحظة: عندما يقيم البرنامج تعابير معقدة والتي تعمل على دمج أنواع مختلفة من المعاملات, فانه يقيم أولا المعاملات الرياضية mathematical operators ، معاملات المقارنة(المضاهاة) comparison operators ثانيا, والمعاملات المنطقية logical operators ثالثا.

يبين الجدول التالي قوائم بأمثلة للمتغيرات المنطقية logical operators عند العمل عليها في التعابير,على افتراض ان المتغير النصي مركبة Vehicle يحتوي على القيمة"Bike" "دراجة", والمتغير العددي الصحيح للسعر يحتوي على القيمة200

Logical expression التعبير المنطقي	Result النتيجة
`Vehicle = "Bike" And Price < 300`	True (both conditions are True) صح(كلا الشرطان صحيحان
`Vehicle = "Car" Or Price < 500`	True (one condition is True) صح (احد الشرطان صحيح
`Not Price < 100`	True (condition is False) صح(لان الشرط خطأ
`Vehicle = "Bike" Xor Price < 300`	False (both conditions are True) خطأ(لان كلا الشرطان صحيحان

تدريب

في التدريب التالي ستعمل على تعديل برنامج(التحقق من مستخدم My User Validation program) بحيث يطلب من المستخدم إدخال personal identification number (PIN) رقم المعرف الشخصي خلال عملية التحقق, ولعمل ذلك,ستعمل على إضافة text box صندوق نص ثاني لتحصل على PIN رقم المعرف الشخصي من المستخدم, ومن ثم تعدل العبارة If...Thenفي تركيب القرار بحيث يستخدم المعامل And للتأكد من رقم المعرف الشخصي PIN

Add password protection by using the *And* operator(إضافة حماية بكلمة مرور وذلك باستخدام المعامل(و

1- عد الى البرنامج السابق واعرض نموذج "التحقق من المستخدم" و أضف *Label*اداة عنوان أخرى لإضافة عنوان جديد الى الفورم وضع هذه الأداة تحت اداة صندوق النص المقنع.

2- ضع ف خاصية النص *Text* لأداة العنوان هذه النص ("رقم المعرف الشخصي PIN")

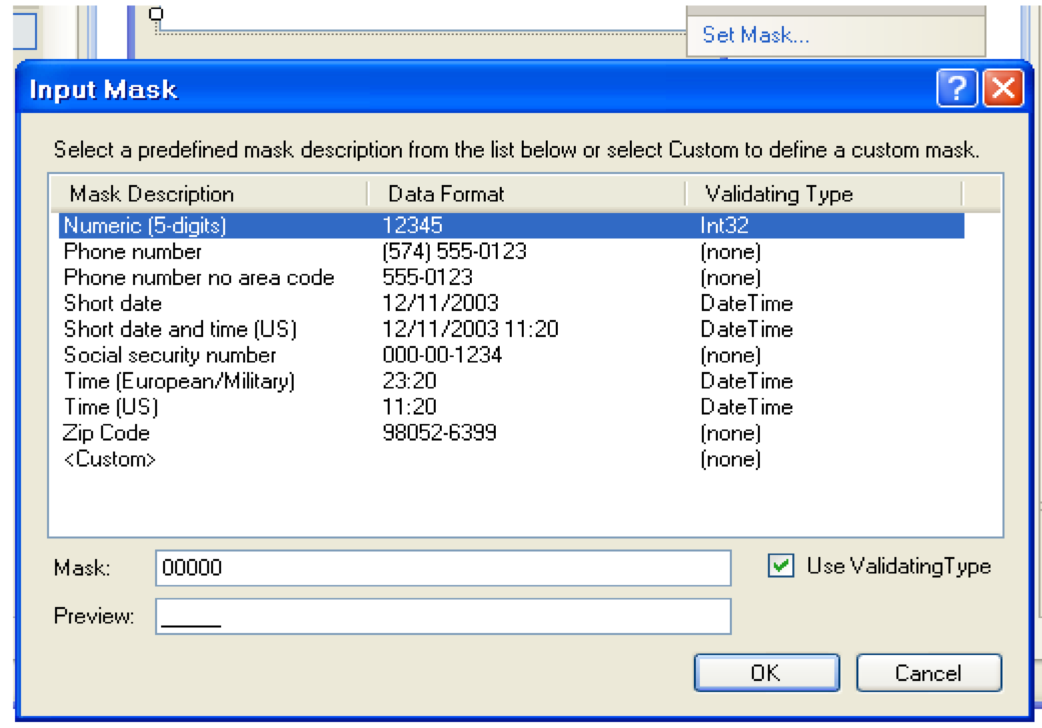
3- أضف اداة *MaskedTextBox*صندوق نص مقنع ثاني الى الفورم تحت اداة العنوان التي أضفتها حديثا.,

4- انقر على shortcut arrow سهم الاختصار الذي على كائن *MaskedTextBox*صندوق النص المقنع لفتح قائمة مهمات صندوق النص المقنع,ومن ثم اضغط الأمر Set Mask ضع قناع لعرض صندوق حوار Input Mask إدخال القناع,



5- اضغط على قناع الإدخال Numeric (5 digits)عددي(5 أرقام) ومن ثم اضغط OK لإغلاق صندوق الحوار هذا.

6- اضغط مزدوج على الزر "التحقق" لفتح إجراء الحدث في محرر الكود,



7- عدل معالج الحدث بحيث يحتوي على الكود التالي:

```
If MaskedTextBox1.Text = "555-55-1212" _
            And MaskedTextBox2.Text = "54321" Then
          MsgBox("ابـحـر بـعـيـداً!")
      Else
          MsgBox("لا استطيع تمييز الرقم")
End If
```

الجملة تحتوي الآن على المعامل المنطقي And,والذي يطلب من المستخدم PINرقم المعرف الشخصي بالتوازي مع رقم التامين الاجتماعي SSNقبل ان يسمح admittedللمستخدم بالدخول الى النظام وفي هذه الحالة الرقم الصحيح للمعرف الشخصي PINهو 54321, (في برنامج حقيقي هذه القيمة سوف يتم استخراجها بالتوافق مع رقم التامين الاجتماعي من قاعدة بيانات الأمن),في الحقيقة ما عملته فقط قمت بإضافة عبارة بسيطة كما يظهر في الكود السابق,حيث عملت على تقطيع الكود الى سطرين باستخدام ميزة التوصيل(الخط المنخفض _) وبعد كتابة هذا الخط قمت بضغط enter لتقطيع سطر الكود وأدخلت عبارة And (و) لتميز الجديد في هذا الكود.

7- اضغط حفظ الجميع من شريط الادوات القياسي ليتم حفظ الإضافات الجديدة.

8- نفذ البرنامج وجرب ان تدخل أرقام مختلفة وتفحص النتائج.

ملاحظة:

تستطيع ان تخصص هذا البرنامج بشكل أكثر فعالية من خلال استخدام خاصية رموز كلمة السر لكائنات صندوق النص المقنع,حيث يمكن استخدام هذه الخاصية لعرض رمز حافظة المكان,مثل النجمة مثلا,عندما يكتب المستخدم,(تستطيع تخصيص الرمز باستخدام نافذة الخصائص) ,ان استخدام ميزة رمز كلمة السر تضفي المزيد من السرية عند يقوم المستخدم بادخال كلمة السر لحماية ميزات خاصة لا يريد احد ان يراها كما تستخدمه دائما في حماية نظام التشغيل خاصتك بكلمة مرور.

ان التنوع في عبارة If. . .Then يمنحك الجملة If. . .Then. . .Else والتي تنفذ بلوك واحد من الجمل اذا كان الشرط صحيح وبلوك آخر من الجمل اذا كان الشرط خاطئً, الشكل العام لجملة If. . .Then. . .Else هو التالي :

```
    If condition Then
        statementblock1()
    Else
        statementblock2()
    End If
```

يقوم الفيجول بيسك الشرط, فإذا كان صحيحاTrue, ينفذ VB البلوك الأول من الجمل ومن ثم يقفز الى العبارات التي تلي العبارة End If.اما اذا كان الشرط غير محقق False فان البيسك سيتجاهل البلوك الأول من الجمل (الذي يلي الكلمة المحجوزة Then) وينفذ البلوك الذي يلي الكلمة المحجوزة Else.

تنوع ثالث لجملة If. . .Then. . .Else والتي تستخدم عدة شروط مع الكلمة المحجوزة ELSEIF

```
If condition1 Then
        statementblock1()
    ElseIf condition2 Then
        statementblock2()
    ElseIf condition3 Then
        statementblock3()
    Else
        statementblock4()
End If
```

تستطيع ان تخصص أي عدد من العبارات ElseIf. والشروط سيتم تقيمها بداية من الأعلى. فإذا ما كان احد هذه الشروط صحيحا فان البلوك الموافق سيتم تنفيذه,العبارة Else اختيارية سيتم تنفيذها اذا ولا واحد من التعابير السابقة صادقة لان يكون صحيحا,المثال التالي يوضح جملة If مع العبارة ElseIf:البرنامج يقوم محصلة الطالب(score) من خلال إدخال المحصلة في صندوق إدخال وبناء على العدد الذي ادخله المستخدم تظهر إحدى الرسائل)Failedفشل, Passمرّ او Very Goodجيد جدا او Excellentممتاز)

```
Dim score As Integer
Dim result As String
    score = InputBox("Enter score")
If score < 50 Then
        Result = "Failed"
    ElseIf score < 75 Then
```

```
        Result = "Pass"
    ElseIf score < 90 Then
        Result = "Very Good"
    Else
        Result = "Excellent"
End If
  MsgBox(Result)
```

ملاحظة: التراكيب (اذا.....عندئذ)المتعددة مقابل (وإلاإذا)Multiple If. . .Then Structures versus ELSEIF

لاحظ انه بعد ان يتم إيجاد الشرط الصحيح فان VB ينفذ الجمل المصاحبة له ويتجاوز عن باقي شروط ELSEIF التي تلي الشرط الصحيح, فيتابع تنفيذ البرنامج بالجمل التي تاتي مباشرة بعد جملة نهاية الشرط إذا(End If)وبالتالي فان الكود ينفذ اكبر نوعا ما.وهذا ما يجعلك تفضل التركيب المعقد مع جمل ELSEIF المستخدمة في المثال السابق عن الجمل في المثال التالي مع ان كلا العبارتين متكافئتين:

```
Dim score As Integer
Dim result As String
score = InputBox("Enter score")
If score < 50 Then
        result = "Failed"
    End If
    If score < 75 And score >= 50 Then
        result = "Pass"
    End If
    If score < 90 And score >= 75 Then
        result = "Very Good"
    End If
    If score >= 90 Then
        result = "Excellent"
End If
  MsgBox(Result)
```

مع جمل if المتعددة سيقوم المترجم بإنتاج كود يتحقق من جميع الشروط حتى ولو كانت المحصلة اقل من 50 .

ترتيب المقارنة هي مسألة حيوية عندما تستخدم جمل ELSEIF متعددة, جرب اكتب الكود السابق بترتيب مختلف كما في المقطع التالي وستحصل على نتيجة غير متوقعة تماما:

```
If score < 75 Then
        result = "Pass"
    ElseIf score < 50 Then
        result = "Failed"
    ElseIf score < 90 Then
        result = "Very Good"
    Else
        result = "Excellent"
End If
```

لنفترض ان المحصلة كانت 49 فان الكود سيقارن المتغير score مع القيمة الاولى 75ولان القيمة 49 تحقق الشرط الأول وهي اقل من75 بالتالي سيسند القيمة (مر)الى النتيجة *Result*,ومن ثم سيتجاوز الشروط المتبقية,وهكذا, فالطالب الذي محصلته 49 سوف يتجاوز الامتحان,لذلك كن حذر جدا وجرب كودك كاملا اذا كان يستخدم الشروط وإلا اذا(ELSEIF) يتوجب عليك إما التأكد من ان الشروط ElseIf تم ترتيبها بطريقة مناسبة أو استخدم الحدود الدنيا و العليا كما في جمل الشرط(إذا If. . .Then)المتعددة.

ملاحظة :الوظيفة The *IIF()* Function

لا تقلق بشان جمل the If. . .Then,يزودك الفيجوال بيسك بالوظيفة IIF() function,وهي وظيفة مبنية داخلياbuilt-in function (يقدمها لك الفيجوال بيسك جاهزة للاستخدام)تقبل كمعامل نسبي تعبير وقيمتين,تعمل على تقيم التعبير وتعود بالقيمة الأولى اذا كان التعبير True صح,أو القيمة الثانية اذا كان التعبير False خطأ و syntax التركيب العام لهذه الوظيفة IIF()هو التالي:

```
IIf(expression, TruePart, FalsePart)
```

المعاملان النسبيان TruePart و FalsePart هما كائنات(يمكن ان يكونا integers عدد صحيح,نصوصstrings, أي كائن سواء built-in معد من اللغة نفسها أو مخصص custom object مثل أنواع البيانات المعرفة من قبل المستخدم كما رايت سابقا).الوظيفة IIF()اكثر إحكاما في تمثيل عبارات الشرط Ifالبسيطة.على فرض انك تريد عرض احد النصوص اما قريب"Close" أو بعيد"Far بالاعتماد على قيمة المتغير (مسافة *distance* variable)فبدل من كتابة شرط Ifمتعدد الجمل, تستطيع استدعاء الوظيفة كمايلي:

```
IIf(distance > 1000, "Far", "Close")
```

مثال نموذجي اخر لوظيفة IIf في تنسيق القيم السالبة وهي نوعا ما شائعة في تطبيقات العمل لعرض كمية سالبة في أقواس, استخدم جملة IIf لكتابة تعبير قصير والذي يعمل على تنسيق الكميات السالبة والموجبة بشكل مختلف, مثل التالي:

```
IIf(amount < 0, "(" & Math.Abs(amount).ToString("#,###.00") & ")",_ amount.ToString("#,## .00"))
```

الطريقة Abs لفئة الرياضيات Math class تعود بالقيمة المطلقة absolute value لقيمة عددية ما,والمعامل النسبي للطريقة ToString يخصص للكمية رقمان عشريان.

اختر حالة Select Case

في فيجوال بيسك, يُمَكِنُك أيضاً التحكم في تنفيذ البيانات في برامجكَ باستعمال تراكيب القرار اختيار الحالة,وتركيب اختيار الحالة مشابه لتركيب If...Then...ElseIf ولكنه أكثر فعالية عندما يعتمد التفرع على مفتاح(كلمة مفتاحيه) رئيسي واحد,او حالة اختيار,تستطيع ايضا استخدام تراكيب اختيار الحالة لجعل كود برنامجك أكثر قابلية للقراءة,

الشكل العام لتراكيب قرار اختيار الحالة هو التالي:

```
Select Case variable
            Case value1
statements executed if value1 matches variableيـتم تـنـفـيـذ هذه الجمل اذا وافقت القيمة1المتغير
            Case value2
statements executed if value2 matches variableتُنفذ في حالة موافقة القيمة2 المتغير
            Case value3
statements executed if value3 matches variableتنفذ في حالة موافقة القيمة3 المتغير
...
            Case Else
statements executed if no match is found تنفذ في حالة عدم مطابقة أي قيمة للمتغير
End Select
```

يبدأ تركيب اختيار الحالة بالكلمة المحجوزة Select Case وينتهي بالكلمة المحجوزة End Select,.طبعا ستستبدل المتغير variable بأي متغير او خاصية,او تعبير أخر يمكن ان يكون القيمة الرئيسية,او أية حالة اختبار للتركيب,.وتقوم باستبدال القيمة value1وvalue2 و value3 بإعداد,او نصوص,او قيم أخرى تتعلق بحالة الاختبار المعتبرة.اذا وافقت احد القيم المتغير,الجمل التي تحت عبارة الحالةCase يتم تنفيذه, ومن ثم يقفز الفيجوال بيسك الى السطر الذي بعد عبارة End Select ويبادر picks up التنفيذ هناك,يمكنك ان

تُضمن أي عدد من عبارات الحالة Caseفي التركيب اختيار الحالةSelect Case, وتستطيع ايضا ان تضمن أكثر من قيمة واحدة في Caseحالة ما,اذا أدرجت قيم متعددة بعد حالة Caseما ,افصل بينهم بالفواصل.commas , يريك المثال التالي كيف يمكن استخدام التركيب Select Case لطباعة الرسالة المناسبة في برنامج ما حول عمر شخص ما والمعالم الثقافية cultural milestones حيث ان متغير العمر يحتوي على القيمة 18,والنص ”!You can vote now“ " تستطيع التصويت الآن" تم إسناده الى خاصية النص Text لكائن اداة العنوان,(ستلاحظ ان " المعالم" لها الطابع الأمريكي American slant , خصص بحرية ما يطابق محيطك الثقافي cultural setting.

ملاحظة:اقرأ النص في الكود والذي يكون باللغة العربية من اليسار الى اليمين

```
Dim Age As Integer
        Age = 18
Select Case Age
            Case 16
                Label1.Text = "تستطيع القيادة الان!"
            Case 18
                Label1.Text = "تستطيع التصويت الان!!"
            Case 21
                Label1.Text = "تستطيع ان تشرب مع الوجبات."
            Case 65
                Label1.Text = "الوقت المناسب لتقاعد وتتمتع بحياتك!"
End Select
```

تركيب Select Case structure اختيار الحالة يدعم ايضا العبارة Case Else والتي يمكن ان تستعملها لعرض رسالة(اذا ولا حالة من الحالات السابقة طابقت المتغير Age),إليك كيف تعمل الحالة(غير ذلك) في المثال التالي,(لاحظ أنني غيرت العمر في المتغير "عمر" الى 25 لأطلق الشرط Case Else "غير ذلك")

```
Dim Age As Integer
        Age = 25
Select Case Age
            Case 16
                Label1.Text = "You can drive now!"
            Case 18
                Label1.Text = "You can vote now!"
            Case 21
                Label1.Text = "You can drink wine with your meals."
            Case 65
                Label1.Text = "Time to retire and have fun!"
            Case Else
                Label1.Text = "You're a great age! Enjoy it!"
End Sele
```

Using Comparison Operators with a *Select Case* Structure " استخدام معاملات المقارنة مع التركيب " اختيار الحالة

تستطيع استخدام معاملات المقارنة لتضمن مجال من قيم الاختبار في التركيب "elect Case اختيار الحالة " .ومعاملات المقارنة للفيجوال بيسك التي يمكن ان تستخدم هي :"يساوي =", و"لايساوي <> " ," اكبر من> " و" اصغر من < " و " =>اكبر او يساوي " و " =<. اصغر او يساوي " لاستخدام معاملات المقارنة تحتاج ان تضمن الكلمة المحجوزة is او الكلمة المحجوزة Toفي التعبير لتحديد المقارنة التي تقوم بها,ترشد instructsالكلمةالمحجوزة Is المترجم لمقارنة المتغير المختبر مع التعبير المدرج بعد الكلمة المحجوزة Is , أما الكلمة المحجوزة To تحدد مجال من القيم. التركيب التالي يستخدم Is, To, والعديد من معاملات المقارنة لاختبار المتغير Ageوعرض واحدة من خمس رسائل.

```
Dim Age As Integer
Select Case Age
            Case Is < 13
                Label1.Text = "Enjoy your youth!"
            Case 13 To 19
                Label1.Text = "Enjoy your teens!"
            Case 21
                Label1.Text = "You can drink wine with your meals."
            Case Is > 100
                Label1.Text = "Looking good!"
            Case Else
                Label1.Text = "That's a nice age to be."
End Select
```

اذا كانت القيم للمتغير Ageاقل من 13 فالرسالة”!Enjoy your youth“ " استمتع بطفولتك" سيتم عرضها . أما من العمر 13 الى العمر19 سيتم عرض الرسالة ”Enjoy your teens!“ " استمتع بمراهقتك" , وهكذا . كما تلاحظ ان تركيب القرار Select Case " اختيار الحالة أوضح من التركيب If...Then " اذا....عندئذ" وأكثر فعالية وخاصة عندما تقوم بعمل ثلاث تفرعات قرار او أكثر بالاعتماد على متغير واحد , او خاصية., ولكن مهما يكن, عندما تعمل مقارنتين او اقل او عندما تعمل مع عدة قيم مختلفة , من المحتمل انك تريد استخدام تركيب القرار If...Then في التمرين التالي,سترى كيف تستطيع استخدام التركيب Select Case لمعالجة المدخلات من اداة صندوق القائمةlist box . ستستخدم الخاصيات: خاصية النص Text وخاصية SelectedIndexChanged "تغيير الفهرس المختار" لأداة صندوق القائمة*ListBox1*, لتحسب المدخلات,ومن ثم سوف تستخدم التركيب *elect Case*لعرض greeting ترحيب في واحدة من اللغات الأربع.

تدريب

استخدام التركيب Select Case لمعالجة المدخلات من صندوق قائمة

1- من قائمة ملف File اضغط New Project مشروع جديد. تلاحظ ان صندوق حوار المشروع الجديد يتم فتحه لك.
2- قم باختيار مشروع Windows Forms Application تطبيق نماذج ويندوز وسمه " اختيار الحالة ", واضغط ok لتفتح لك الفورم الفارغ في المصمم
3- من صندوق الادوات toolbox اختار Label اداة عنوان اضغط عليها وقم بسحبها حتى تصبح فوق الفورم ثم قم بتركها) ومن ثم قم بنقلها الى أعلى الفورم بالفارة وبعملية السحب والترك العادية وهذه الأداة سنستخدمها كعنوان للبرنامج,وبنفس الطريقة قم بإضافة Label اداة عنوان أخرى الى الفورم وضعها تحت الأداة الأولى والتي سنستخدمها كعنوان لأداة ListBox صندوق القائمة.
4- اضغط على اداة صندوق القائمة ListBox في صندوق الادوات وقم بسحبها كما فعلنا سابقا,وضعها تحت اداة Label العنوان الثانية .
5- أضف ايضا Label أداتي عنوان تحت صندوق النص وسنستخدمها لعرض مخرجات البرنامج.
6- أضف Buttonاداة الزر منtoolbox صندوق الادوات لإنشاء زر Buttonعلى الفورم form كما فعلنا سابقا,.
7- افتح Propertiesنافذة الخصائص ومن ثم قم بإعداد الخاصيات التالية للأدوات التي قمت بانشاءها على الفورم:

Object الكائن	Property الخاصية	Setting الاعدادات
Form1	Text	" حالة الترحيب "
Label1	Font	Times New Roman, Bold, 12-point
	Name	lblTitle
	Text	برنامج الترحيب العالمي
Label2	Name	lblTextBoxLabel
	Text	" اختار بلد "
Label3	Font	10-point
	Name	lblCountry
	Text	(اتركه فارغ)
Label4	AutoSize	False
	BorderStyle	Fixed3D
	ForeColor	Red
	Name	lblGreeting
	Text	(empty)
ListBox1	Name	lstCountryBox
Button1	Name	btnQuit
	Text	"خروج"

عندما تنتهي من الإعدادات فان الفورم تبدو مشابهة للمعروضة هنا:



الآن ستدخل كود البرنامج لإسناد القيم الأولية لصندوق الأدوات:

8- اضغط مزدوج في أي مكان على الفورم(انتبه أن لا تضغط على أي أداة من الأدوات الموجودة عليها) ليفتح لك محرر الكود والذي ينقلك مباشرة إلى داخل الإجراء تحميل الفورم Form1_Load

9- اكتب الكود التالي في من اجل التمهيد لصندوق القائمة:

```
lstcountrybox.Items.Add("سوريا")
lstcountrybox.Items.Add("مصر")
lstcountrybox.Items.Add("فلسطين")
lstcountrybox.Items.Add("السعودية")
```

These lines use the Add method of the list box object to add entries to the list box on

تستخدم هذه الأسطر الطريقة Addلكائن صندوق النص list boxلإضافة إدخالات إلى صندوق القائمة على الفورم :

10- اضغط العروة [Design] Form1.vb عند أعلى محرر الكود لتنتقل راجعا إلى المصمم, ومن ثم اضغط مزدوج على كائن صندوق القائمة من على الفورم لتحرير إجراء حدثه:

إجراء الحدث lstcountrybox_SelectedIndexChanged يظهر في محرر الكود

11- اكتب السطور التالية لمعالجة قائمة اختيار الصندوق الموضوعة بواسطة المستخدم:

```
lblcountry.Text = lstcountrybox.Text
        Select Case lstcountrybox.SelectedIndex
```

```
            Case 0
                lblgreeting.Text = "بكم تـرحب سوية"
            Case 1
                lblgreeting.Text = "تحيكم مصر"
            Case 2
                lblgreeting.Text = "تنتظركم فلسطين"
            Case 3
                lblgreeting.Text = "معكم السعودية "
End Selec
```

السطر الأول ينسخ اسم البند المختار من صندوق القائمة list box إلى خاصية النص Textالتابعة إلى label أداة العنوان الثالثة على الفورم,(والتي غيرت خاصية الاسم name(لها) إلى الاسم lblCountry).الخاصية الأكثر أهمية التي تم استخدامها في العبارات هي lstcountrybox.Text والتي تحوي على نص البند تماما الذي تم اختياره في صندوق القائمة,أما العبارات الباقية هي جزء من تركيب قرار اختار الحالة.,التركيب يستخدم الخاصية lstcountrybox.SelectedIndex كمتغير لحالة اختيار ويقارنها مع عدة قيم. البند SelectedIndex" الفهرس المختار" هو 0 ,البند التالي 1 وهكذا,. باستخدام SelectedIndex الفهرس المختار فان التركيب Select Caseاختيار الحالة يستطيع وبسرعة تحديد اختيار المستخدم وعرض رسالة الترحيب الصحيحة على الفورم

12- اعرض الفورم مرة ثانية واضغط مزدوج على زر button" خروج" ليفتح لك إجراء الحدث btnQuit_Click في محرر الكود

13- اكتب الكلمة المحجوزة end" نهاية" في إجراء الحدث.

```
end
```

14- اضغط الزر Save All"أحفظ الجميع " من على شريط الأدوات القياسي Standard toolbar لحفظ كل التغيرات التي عملتها,حدد مكان مجلد الحفظ كمكان لحفظ المشروع.

الآن نفذ المشروع ,وشاهد كيف تعمل عبارة Select Case" اختار الحالة "

جرب الانتقال من بلد لأخر في صندوق القائمة ولاحظ كيف تظهر النتائج مباشرة في أداتي العنوان الثالثة والربعة حيث أن أداة العنوان الثالثة تعرض اسم البلد الذي اخترته من صندوق القائمة أما أداة العنوان الرابعة فإنها تعرض رسالة الترحيب الخاصة بكل بلد, بعد أن تنتهي من تنفيذ المشروع وتلاحظ التغيرات التي تتم, اضغط على الزر خروج لإنهاء البرنامج والعودة إلى بيئة التطوير مرة أخرى.

خطوة أخرى إلى الأمام: التحري عن أحداث الفارة:One Step Further: Detecting Mouse Events

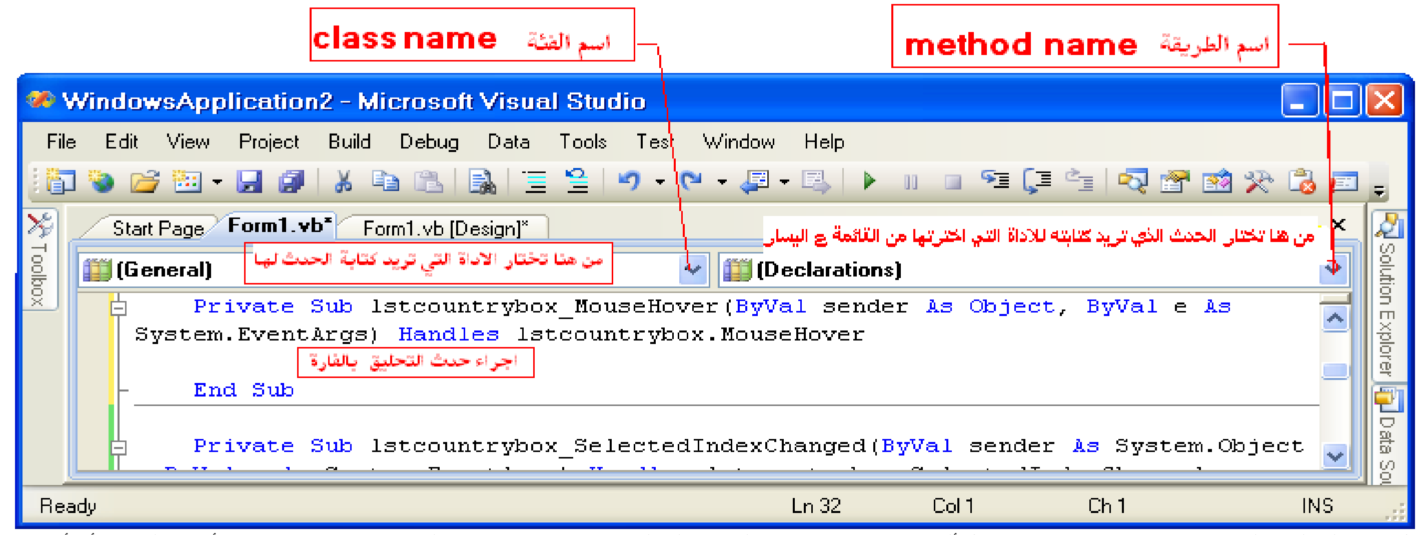
كما رأيت سابقا فقد مر معنا العديد من الأحداث التي يمكن أن تستجيب لها برامج الفيجوال بيسك. ومع تقدمنا في الفصل تعلمت كيف تدير الأنواع المختلفة من الأحداث,باستخدام تراكيب القرار *If...Then* و *Select Case* في هذا المقطع, *ستعمل على إضافة معالج حدث ما لبرنامج "Select Case" اختر حالة" والذي يظهر عندما* hovers *يحلق المؤشر فوق صندوق قائمة البلد لهنيهة ,لتستنبط إجراء خاص, أو معالج حدث صندوق القائمة من اجل حدث تحليق الفار ةMouseHover* , واحد من العديد من الفعاليات المتعلقة بالفارة , والتي يمكن للفيجوال بيسك أن يظهرها ويعالجها.ومعالج الحدث هذا سيعرض الرسالة "من فضلك انقر على اسم البلد" إذا ما أشار المستخدم إلى صندوق قائمة البلد لحظة وقبل أن يقوم بعملية الاختيار.ربما بسبب عدم معرفته كيف يقوم بعمل اختيار أو انه engrossed منهمك في قضية أخرى.

إضافة معالج حدث للفارة:

1- افتح محرر الكود إذا لم يكن مفتوح مسبقا.

2- عند أعلى محرر الكود,اضغط على سهم اسم الفئة.ومن ثم اضغط على الكائن ليتم اختياره في هذه القائمة. تستطيع استخدام feature ميزة أداة النصحToolTip للمساعدة في تحديد العناصر مثل صندوق قائمة اسم الفئة في الفيجوال استوديو,والتي هي في مثال لأخر عن *MouseHover*حدث تحليق الماوس ضمن بيئة تطوير المتكاملة IDE

3- اضغط على سهم اسم الطريقة في القائمة المجاورة على يمين القائمة السابقة ومن القائمة المنسدلة التي تظهر قم باختيار الحدث "تحليق الفارة *MouseHover*" , ترى أن الفيجوال بيسك يفتح إجراء حدث تحليق الفارة لصندوق قائمة البلد*lstCountryBox_MouseHover* في محرر الكود كما في الشكل التالي :



كل كائن على الفورم لديه إجراء حدث واحد مميز يفتح بشكل آلي عندما تضغط مزدوج عليه من على الفورم, إذا كنت بحاجة مثلا لبرمجة إجراء حدث ما من الأحداث المتبقية لأي أداة تم وضعها على الفورم عليك أن تستخدم Method Name صندوق قائمة اسم الطريقة كما هو بيناه في الشكل السابق

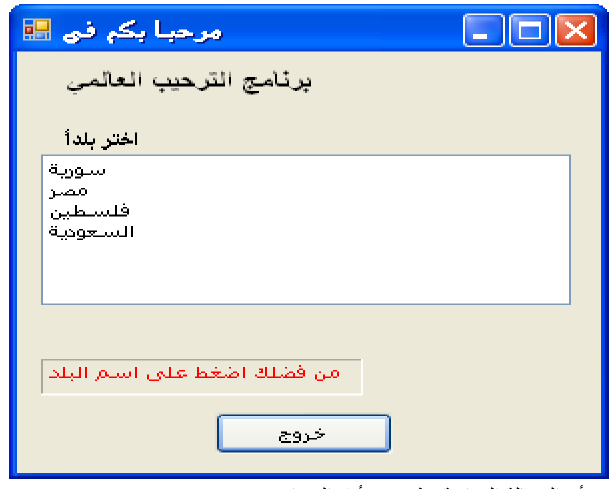
4- اكتب العبارات البرمجية التالية في إجراء الحدث lstcountrybox_MouseHover:

```
If lstcountrybox.SelectedIndex < 0 Or lstcountrybox.SelectedIndex > 4 Then
        lblgreeting.Text = "من فضلك اضغط على اسم البلد"
End If
```

عبارة *If* تقيم الخاصية لكائن صندوق القائمة باستخدام جملتي شرطي مع المعامل *Or*,يفترض معالج الحدث أنه إذا كان هناك قيمة بين 1و4 في الخاصية *SelectedIndex* فان *المستخدم لايحتاج المساعدة في معرفة اسم البلد (والذي قام باختيار بلد ما مسبقا) ولكن إذا كانت الخاصية SelectedIndex خارج المجال فان معالج الحدث يعرض رسالة* من فضلك اضغط على اسم البلد في أداة عنوان التحية عند أسفل الفورم، تظهر رسالة المساعدة هذه عندما يحتفظ المستخدم بالمؤشر فوق صندوق القائمة والتي تختفي عندما يختار المستخدم اسم بلد ما .

5- اضغط على الزر ابدأ التصحيح ليتم تشغيل البرنامج Start Debugging

6- حافظ على المؤشر فوق صندوق قائمة البلد وانتظر للحظات، ستظهر لك الرسالة " من فضلك اضغط على اسم البلد " باللون الاحمر للنص في أداة العنوان كما هو مبين في الشكل



7- اضغط على اسم البلد في صندوق القائمة,تشاهد أن الرسالة الموافقة تظهر ف أداة العنوان.
8- اضغط على زر الخروج لإنهاء البرنامج.

لقد تعلمت كيف تعالج أحداث الفارة في البرنامج وتعلمت أيضا أن كتابة معالج الحدث هي عملية بسيطة تماما, كبديل فعال لتركيب ElseIf لجمل متعددة ولكن أكثر صعوبة في قراءة الكود هو التركيب Select Case "اختر حالة" والذي يعمل على مقارنة التعبير نفسه ولكن بقيم مختلفة,الفائدة من الجملة "اختر حالة" عوضا عن الجمل ElseIf. . .Then. . .If. هي في أنها تجعل الكود أسهل في عملية القراءة والتصحيح,. تركيب اختيار الحالة Select Case يقيم تعبير مفرد فقط ويوضع هذا التعبير عادة في أعلى التركيب,ونتيجة التعبير expressionيتم مقارنتها بعدئذ مع قيم متعددة، فإذا ما وافقت النتيجة واحدة من هذه القيم فان مقطع الجمل الموافق سيتم تنفيذه,إليك الشكل العام لجمل Select Case "اختر حالة"

```
Select Case expression
        Case value1
          statementblock1()
        Case value2
          statementblock2()
          .
          .
        Case Else
          Statementblock(N)
End Selec
```

واليك المثال العملي التالي:

```
Dim Message As String
     Select Case Now.DayOfWeek
        Case DayOfWeek.Monday
          Message = "Have a nice week"
        Case DayOfWeek.Friday
          Message = "Have a nice weekend"
        Case Else
          Message = "Welcome back!"
End Select
     MsgBox(Message)
```

في هذه القائمة التعبير الذي يتم تقيمه في بداية الجملة هو الطريقةNow.DayOfWeek, هذه الطريقة ترجع بمكونات العداد DayOfWeek " أيام الأسبوع" وتستطيع استعمال أسماء هذه المكونات في كودك لجعله أسهل للقراءة,وقيمة هذا التعبير يتم مقارنتها مع القيم التي تتبع كل كلمة محجوزةCase, فإذا ما تمت المطابقة مع أي من هذه القيم فان بلوك الكود المرافق لهذه القيمة يتم تنفيذه,ويتجاوز البرنامج باقي القيم في عبارة Case وينتقل لينفذ الجمل التي تلي End Selectإنهاء الاختيار,فالحالتين الأولى والثانية من عبارات الحالة تأخذ في حسبانها يوم الاثنين والجمعة على الترتيب.أما الجملةCase Else تأخذ في حسبانها باقي أيام الأسبوع,بعض جمل الحالة Caseيمكن أن يتم إتباعها بقيم متعددة,والتي تفصل بينها فواصل كما في التعبير التالي,

```
Select Case Now.DayOfWeek
        Case DayOfWeek.Monday
          Message = "Have a nice week"
        Case DayOfWeek.Tuesday, DayOfWeek.Wednesday, DayOfWeek.Thursday
          Message = "Welcome back!"
        Case DayOfWeek.Friday, DayOfWeek.Saturday, DayOfWeek.Sunday
          Message = "Have a nice weekend!"
End Select
     MsgBox(Message)
```

كما ترى تمت عملية معالجة قيم متعددة بعد عبارة الحالة Caseوذلك بالفصل بين هذه القيم فقط بفواصل,وهذا التركيب لا يحتوي على عبارة الحالة " وإلا " Case Else لان جميع القيم المكنة تم اختيارها هنا في عبارات الحالةCase, والطريقة DayOfWeek لا يمكن لها أن تعود بقيم أخرى.
يمكن أن يتم تعقيد عبارة الحالة Case أكثر من ذلك قليلا,فمثلا يمكنك أن تميز حالة ما بحيث يكون المتغير اكبر(أو اصغر) من قيمة ما, لتعالج هذا المنطق, استخدم الكلمة المحجوزة Is كما في مقطع الكود التالي الذي يميز بين النصف الأول والنصف الثاني من الشهر:

```
Select Case Now.Day
        Case Is < 15
          MsgBox("It's the first half of the month")
        Case Is >= 15
          MsgBox("It's the second half of the month")
 End Select
```

قصر الدارة باستخدام(وايضا *AndAlso*)و(أو وإلا *OrElse*)

توفر الفيجوال بيسك معاملين منطقيين تستطيع ان تستخدمهما في جملك الشرطية,هما المعاملان (وايضا AndAlso)و(أووالا OrElse) وهذان المعاملان يقومان بنفس العمل الذي يقوم به المعاملان And وOr على الترتيب respectively, ولكنهما يوفران مرونة هامة جدا subtlety في الطريقة التي يتم فيها تقيم كل منهما,وهذا جديد على المبرمجين المحترفين بالفيجوال بيسك6, اعتبر ان جملة If لديها حالتين (شرطيين) وهما مرتبطان بواسطة المعامل AndAlso.فمن اجل ان يتم تنفيذ جمل التركيب, كلا الشرطين يجب ان يتم تقيمه الى صح ,فإذا ماتم تقيم الشرط الأول الى خطأ,فان فيجوال بيسك يتجاوز مقطع الكود المرافق له ويذهب مباشرة الى السطر الذي يليه او الى عبارة (Else أخر) مباشرة,بدون ان يختبر الشرط الثاني , هذه التقييم الجزئي او القصر في دارة لجملة يكِّون مفهوم منطقي يقول التالي:لماذا يجب ان يستمر الفيجوال يسك في تقيم كلا الشرطين في جملة If اذا لم يكن كلا الشرطين صحيحين؟

المعامل OrElse يعمل بنفس الأسلوب,اعتبر ان عبارة If لديها شرطان بحيث يكونان مرتبطان بواسطة المعامل OrElse,فمن اجل ان يتم تنفيذ تركيب If,على الأقل شرط منهم يجب ان يتم تقيمه الى صح ,فإذا ما تم تقيم الشرط الأول الى صح يبدأ الفيجوال بيسك بتنفيذ الجمل في تركيب If مباشرة immediately, بدون اختبار الشرط الثاني,إليك المثال التالي لحالة short-circuit قصر الدارة في الفيجوال بيسك:الإجراء البسيط simple routine الذي يستخدم جملة If والمعامل AndAlso لاختبار شرطين وعرض رسالة"Inside If" "من داخل اذا" اذا كان كلا الشرطان صحيح.

```
Dim Number As Integer = 1
        If Number = 1 AndAlso MsgBox("Second condition test") Then
            MsgBox("إذا عبارة داخل من")
        Else
            MsgBox("اخر عبارة داخل من")
        End If
```

الوظيفة صندوق الرسالة MsgBox نفسها يتم استخدامها كـاختبار حالة(شرط) ثاني,والتي تبدو نوعا ما غير عادية,ولكن هذا التركيب الغريب ممكن تماما,ويعطينا نتيجة مثالية لرؤية كيف تتم عملية قصر الدارة بشكل أكثر قربا وفهما,النص"اختبار الشرط الثاني" يظهر في صندوق رسالة فقط اذا تم إسناد المتغير العددي `Number` الى 1,أما في الحالات الأخرى فان المعامل AndAlso يقصر دارة الجملة,والشرط الثاني لم يتم تقيمه,.اذا جربت هذا الكود بشكل عملي,تذكر ان هذا فقط لتوضيح الفكرة ,فلن تستخدم صندوق النص بهذا الشكل كشرط حيث انه في حقيقة الأمر لا يختبر شيء,ولكن بتغير متغير العدد `Number` من 0 الى 1 وبالعكس، تستطيع ان تحصل على فكرة جيدة عن كيفية عمل المعامل AndAlso وقصره الدارة.

إليك مثال ثاني عن وظائف قصر الدارة في الفيجوال بيسك,عندما يتم تقييم شرطين باستخدام المعامل *AndAlso*,هذه المرة اختبار شرطي أكثر تعقيدا(7/عمر الإنسان<=1)هذا الشرط يتم استخدامه بعد المعامل *AndAlso* لتحديد ما يسميه بعض الناس "عمر الكلب" لشخص ما.

```
Dim HumanAge As Integer
        HumanAge = 7
        'سنة الكلب تساوي سبع سنوات بشرية
        If HumanAge <> 0 AndAlso 7 / HumanAge <= 1 Then
            MsgBox("عمرك على الأقل سنة كلب واحدة ")
        Else
            MsgBox("عمرك اقل من سنة كلب واحدة")
        End If
```

وهذا جزء من برنامج(ضخم هكذا يسمى عمر الكلب لشخص ما),وذلك بتقسيم عمره على 7 ,وهذا الإجراء يحاول تحديد فيماذا كانت القيمة في المتغير `HumanAge` على الأقل 7 سنوات(اذا لم تكن قد سمعت بمبدأ "عمر الكلب"من قبل , فإليك ما يعني اذا كان عمر شخص ما 28 سنة,ستكون 4 سنوات بعمر الكلب, وهذا أقترح كطريقة مثيرة تتَعَلُّق بالكلاب، حيث أن للكلابِ lifespan فترة حياة تقريباً roughly سُبع التي للبشر.). يَستعملُ الكود شرطين في جملة If ويُمْكِنُ أَنْ يُستَعملَ في تَشْكِيلة مختلفة من السياقاتِ يمكنك ان تستعمله في إجراءِ حدثِ النقر على كائن زرّ ما. يُدقّقُ الشرطُ الأول لرُؤية فيم اذا تم وضع العدد صفر في المتغير HumanAge, افترضت بشكل مؤقت بأنّ المستعملَ عِنْدَهُ بما فيه الكفاية من الحس لوَضْع عُمر إيجابي في المتغير HumanAge لأن عدد سلبي يُنتجُ النَتائِجَ الخاطئة. الشرط الثاني يَختبرُ فيم اذا الشخصَ بعمر سبعة سنوات على الأقل.إذا ما تم تقيم كلا الشرطين إلى الصواب،يتم عرض الرسالة `"عمرك على الأقل سنة كلب واحدة "` في صندوق رسالةِ. إذا كان الشخص أقل مِنْ سبعة، يتم عرض الرسالة `"عمرك اقل من سنة كلب واحدة"`.

تخيّل الآن بأنّني غيّرتُ قيمة المتغيّرِ HumanAge مِنْ 7 إلى 0. ماذا يَحْدثُ؟ شرط الجملة If يتم تقيمه الى خطأ من قبل مترجم فيجوال بيسكَ،وذلك التقييم يَمْنعُ الشرط الثاني من أنْ يُقيّمَ، هكذا الإيقاف halting ، أو الاختصار short-circuiting. ايضا جمل If تحمينا من nasty الشذوذ " التقسيم على الصفر" الذي يُمْكِنُ أَنْ يَنْتجَ إذا قسّمنَا 7 على الـ0 (القيمة الجديدة للمتغيّرِ HumanAge). فلم يكن عِنْدَنا نفس الإمكانية في فيجوال بيسكِ 6. فوضعُ المتغيّرُ HumanAge إلى 0 في فيجوال بيسكِ 6 كَانَ سَيُنتجُ عنه خطأ وقتِ تشغيل و انهيار كامل للبرنامج، لأن كامل جملة(إذا If) كان يُمكنُ أنْ تقيّمَ، والقسمة على صفر غير مسموح به permitted في فيجوال بيسكِ 6. في الفيجوال الأستوديو حصلنا على فائدة مِنْ سلوكِ الاختصار short-circuiting .

الخلاصةِ، يَفْتحُ المعاملان AndAlso وو OrElse في فيجوال بيسكِ بضعة إمكانيات جديدة لمبرمجي فيجوال بيسكِ،من ضمنها الإمكانيةِ لمَنْع أخطاءِ وقتِ التشغيل والنَتائِج الأخرى الغير متوقّعة. ومن المحتملُ أيضاً تحسين الأداءِ بوَضْع تلك الشروطِ التي تستهلك وقت للحِساب في نهايةِ جملة الشرط، لأن فيجوال بيسكَ لا يُؤدّي حساباتِ الشرطِ هذه المكلفة مالم تكن ضرورية. على أية حال، تحتاج لان تفكر بعناية حول كُلّ الشروط المحتملة التي قَدْ تُصادفُها جملة If عند تغيير حالات المتغير أثناء تنفيذ البرنامج.

يوجد شرك(مأزق pitfall)عام في تقيم التعابير في الفيجوال بيسك وهو محاولة مقارنة القيمة لاشيء Nothing بشيء ما,(فالمتغير الكائني الذي لم يتم إسناد أي قيمة له لا يمكن استخدامه في الحسابات أو في المقارنات) خذ الجمل التالية:

```
Dim B As SolidBrush
    B = New SolidBrush(Color.Cyan)
    If B.Color = Color.White Then
        MsgBox("Please select another brush color")
    End If
```

هذه الجمل تنشئ من المتغير الكائني SolidBrush المتغير B, ومن ثم تختبر لون الفرشاة brush ,وتمنع prohibit المستخدم من استخدام اللون الابيض في الرسم, في الجملة الثانية من مقطع الكود تم التمهيد للمتغير B إلى اللون "السيان" فكل شكل يتم رسمه بهذه الفرشاة يظهر بلون السيان, فإذا حاولت أن تستخدم المتغير B بدون أن تمهد له,سيؤدي إلى حدوث خطأ وقت التنفيذ,وهذا الخطأ البشع"استثناء NullReferenceException "بدون قيمة" مرجعية",في مثالنا استثناء(خطأ وقت التنفيذ)سيتم إطلاقه عندما يدخل البرنامج إلى جملة الشرط,لان المتغير ليس لديه(انه لاشيء) والكود يحاول مقارنة something شيء ما بـقيمة لاشيء Nothing والتي لا يمكن مقارنتها إلى أي شيء, لتجرب ذلك اجعل العبارة الثانية من الكود تعليق وذلك بوضع علامة اقتباس مفردة أمام `B = New SolidBrush(Color.Cyan)` لجعلها تعليق,ومن ثم قم بتنفيذ البرنامج لترى ما يحدث ومن اعمل على إزالة علامة الاقتباس لإرجاع السطر إلى الكود مرة ثانية, دعنا نصلح هذا ,وذلك من خلال التأكد من أن المتغير B ليس(لاشيء)

```
If B IsNot Nothing And B.Color = Color.White Then
        MsgBox("Please select another brush color")
End If
```

عبارة If يجب أن تقارن خاصية Color اللون للكائن B, فقط اذا كان لهذا الكائن قيمة(وليس لاشيء Nothing), وهذه الحالة غير مطبقة هنا,فالمعامل AND"و" يقيم جميع الشروط في التعبير ومن ثم يدمج نتائجها(قيم صح أو خطأ) لتحديد قيمة التعبير.فإذا كانت جميع الشروط صحيحة فان النتيجة صح , ومهما يكن فانه لن يتجاوز عن تقيم بعض الشروط ولو كانت خطأ ,لتجنب المقارنات غير الضرورية, استخدم المعامل AndAlso"بالإضافة إلى" . المعامل AndAlso يعمل ما يجب أن يعمله المعامل "و" في الطرف الأول من التعبير أي يعمل على مقارنة الشرط الأول فإذا كان خطأ فلا يوجد حاجة لمقارنة الشرط الثاني, أي أن المعامل B اذا كان لاشيء,فلا يوجد سبب لمتابعة مقارنة لونه,وكامل التعبير سيتم تقيمه إلى خطأ,بغض النظر عن لون الفرشاة, إليك كيفية استخدام المعامل AndAlso

```
If B IsNot Nothing AndAlso B.Color = Color.White Then
      MsgBox("Please select another brush color")
End If
```

يقال عن المعامل AndAlsoانه قصر التقييم لكامل التعبير حالما أتى في مساره قيمة خاطئة, فحالما ينقلب جزء في عملية AndAlsoإلى خطأ فكامل التعبير خاطئ ولا توجد حاجة لتقييم الشروط المتبقية,. يوجد معامل أخر مكافئ لقصر دارة التعبير للمعاملOR وهو المعامل OrElse, هذا المعامل يمكن أن يعمل على زيادة سرعة عملية المقارنة للتعابير المنطقية نوعا ما وهذا المعامل ليس بأهمية المعامل AndAlso,يوجد سبب أخر جيد من اجل قصر تقيم التعبير وهو للمساعدة في إتمام العمل.فإذا ما اخذ الشرط الثاني للمعامل "و" وقت طويل لتنفيذه(مثلا عليه التمكن من الوصول إلى قاعدة بيانات بعيدة) تستطيع استخدام المعامل AndAlsoلجعل التأكد من عدم تنفيذ الشرط الأخر عندما يكون لا حاجة له.

جمل الدوران Loop Statements:

تسمح لك عبارات الدوران بتنفيذ واحد أو أكثر من اسطر الكود بشكل متكرر repetitively, العديد من المهمات المؤلفة من العمليات التي يجب أن يتم اعادتها مرة بعد مرة, وعبارات الدوران جزء هام من أي لغة برمجة,وتدعم الفيجوال بيسك جمل الدوران التالية:

- For. . .Next (من .... التالي)
- Do. . .Loop (اعمل......كرر)
- While. . .End While (بينما......نهاية بينما)

اولا- من....التاليFor...Next

بشكل مختلف عن باقي التكرارات الأخرى يتطلب الدوران the For. . .Nextأن تعرف عدد مرات التكرار التي ستقوم بها الجمل الموضوعة ضمن التكرار الذي سيتم تنفيذه, الدوران(التكرار) لديه الشكل العام التالي:

```
For counter = start To end [Step increment]
      statements()
Next [counter]
```

counter = start(بداية العداد) To end(إلى النهاية) [Step increment](خطوة الزيادة) statements(العبارات التي تنفذ داخل التكرار) Next [counter](العداد التالي)

الكلمات المحجوزة في الأقواس المربعة اختيارية(بدك تكتبها أو لا ما في مشكلة) المعاملات النسبية,counter, start, end,والزيادة increment جميعها قيم عددية,يتم تنفيذ التكرار For. . .Next عدد من المرات المطلوبة في العداد counter حتى الوصول(أو تجاوز) القيمة النهائية في حلقة التنفيذ For. . .Next, يعمل فيجوال بيسك التالي:

1- يضع العدادcounter مساوي للبدايةstart
2- يختبر فيما اذا العداد اكبر من النهاية(القيمة العظمى للعداد) اذا كانت كذلك فانه يخرج من الحلقة بدون أن ينفذ الجمل في جسم الدوران.ولا حتى مرة واحدة,فإذا كانت الزيادة سالبة,يختبر البيسك لرؤية فيما اذا العداد اقل من النهاية(القيمة الصغرى) اذا كانت كذلك فانه يخرج من الحلقة بدون ايضا أن ينفذ الجمل في جسم التكرار.
3- ينفذ الجمل في البلوك(المقطع من الكود في جسم الدوران)
4- يزيد قيمة العداد بنفس الكمية المحددة في معامل الزيادة النسبي increment الذي يلي الكلمة المحجوزة Step, فإذا كان معامل الزيادة النسبي *increment*غير محدد, يتم زيادة العداد بالقيمة 1 أما اذا كانت الخطوة قيمة سالبة, ينقص decreasedالعداد counter تبعا لها
5- يستمر بالخطوة رقم 3

الحلقة في القائمة التالية تمسح كل العناصر لبيانات المصفوفة العددية data وتحسب المتوسط لهم:

```
Dim data(50) As Integer
Dim i As Integer, total As Double
    For i = 0 To Data.GetUpperBound(0)
      total = total + Data(i)
    Next i
Debug.WriteLine(total / Data.Length)
```

الشيء الوحيد الأكثر أهمية الذي يجب أن تحفظه في ذاكرتك,عندما تعمل مع الحلقاتFor. . .Next هو أن قيمة النهاية يتم وضعها في بداية التكرار(يجب أن تضع نهاية للتكرار بعد إلى(To)), تغيير قيمة نهاية المتغير في جسم الحلقة لن يعطي ثماره,مثلا.التكرار التالي سيتم تنفيذه عشر مرات وليس مئة مرة:

```
Dim endValue As Integer = 10
    Dim i As Integer
    For i = 0 To endValue
      endValue = 100
  { more statements }
    Next i
```

تستطيع ومهما يكن أن تعدل قيمة العداد *counter* من ضمن الحلقة, العبارة التالية هي مثال عن حلقة غير منتهية(لانهاية من الدوران(التكرار)

```
For i = 0 To 10
      Debug.WriteLine(i)
      i  = i - 1
Next i
```

هذه الحلقة ليس لها نهاية, في الواقع لا تزيد ابد (اذا جربت هذه الحلقة اضغط Ctrl + Breakلقطع الحلقة الغير منتهية)

ملاحظة :لا تعالج عداد الحلقة :

معالجة العداد للحلقة غير مرغوب وبقوة,وهذا النوع مت التدريب على الأغلب سيقود إلى أخطاء مثل الحلقات اللانهائية infinite loops وتجاوز الحدoverflows, وهكذا, اذا كان لا عدد التكرار لحلقة ما,غير معروف مقدما,استخدم الحلقة(اعمل......كررDo. . .Loop) أو الحلقة(بينما...نهاية بينماWhile. . .End While)لتقفز خارج الحلقة قبل الأوان prematurely (قبل أن يتم الانتهاء من التكرار)استخدم العبارة Next For

المعامل النسبي للزيادة*increment* يمكن أن يكون موجب أو سالب,اذا كانت البداية اكبر من النهاية فان قيمة الزيادة يجب أن تكون سالبة,وإلا فان جسم الحلقة لن يتم تنفيذه أبدا,ولا حتى مرة واحدة ,. يسمح لك الفيجوال بيسك 2008 بالإعلان عن عداد من ضمن عبارة الحلقةFor ,. يتوقف متغير العداد من اجل الخروج عندما تحرر الحلقة قيمة ما bails out خارج الحلقة,

```
For i As Integer = 1 To 10
      Debug.WriteLine(i.ToString)
    Next
      Debug.WriteLine(i.ToString)
```

تم استخدام المتغير i كعداد للحلقة,وهو غير مرئي خارج نطاق الحلقة,العبارة الاخيرة لن يتم ترجمتها ومحرر الكود سيضع تحتها خط متعرج ويولد رسالة خطأ"الاسم i لم يتم الاعلان عنه " the Name 'i' is not declared

ثانيا- الحلقة (أعمل...كرر)***Do...Loop***

تنفذ الحلقة (أعمل..كرر)مقطع من العبارات طالما أن شرط ما محقق,أو حتى يصبح شرط ما محقق,تقيم الفيجوال بيسك تعبير ما(شرط الحلقة).فاذا كان صحيحا,فان العبارات في جسم الحلقة يتم تنفيذها, يتم تقييم التعبير أما في بداية الحلقة (قبل تنفيذ أي جملة) أو عند نهاية الحلقة (بلوك الجمل يتم تنفيذه مرة على الاقل).فاذا كان التعبير خطأ,يتابع البرنامج التنفيذ مع العبارات التي تلي الحلقة.يوجد نوعين لجملة Do. . .Loop وكلاهما يستخدم نفس التشكيل الأساسي basic model, يمكن للحلقة أن تنفذ إما بينما يكون الشرط صحيحا, أو حتى يصبح الشرط صحيحا,هذين النوعين يستخدمان الكلمة المحجوزة WhileوUntil لتحديد إلى متى ستبقى العبارات قيد التنفيذ,.
لتنفيذ مقطع من جمل بينما يكون الشرط صحيحا استخدم الشكل العام التالي:

```
Do While condition
        statement(-block)
Loop
```

لتنفيذ مقطع من جمل حتى يصبح الشرط صحيحا استخدم الشكل العام التالي:

```
Do Until condition
        statement(-block)
Loop
```

عندما ينفذ الفيجوال بيسك هذه الحلقات يقيم اولا condition الشرط,اذا كان الشرط خطأ, يتم تجاوز الحلقة Do. . .While (بلوك الحلقة لا ينفذ ولا حتى مرة واحدة)ولكن الحلقة Do. . .Until. يتم تنفيذها, عندما يصل التنفيذ الى عبارة Loop كرر,يقيم الفيجوال بيسك التعبير مرة ثانية,يكرر بلوك الجمل للحلقةDo. . .While, اذا كان التعبير(الشرط) ما يزال صحيحا,او يكرر الجمل للحلقة Do. . .Until اذا كان التعبير خطأ, باختصار,الحلقة Do. . .While يتم تنفيذها عندما يكون الشرط صح والحلقة Do Until يتم تنفيذها عندما يكون الشرط خطأ,.اما النوع الأخير لجملة Do. . .Loop يسمح لك بتقييم الشرط لدى نهاية التكرار,إليك الشكل العام لكلا الحلقتين,مع تقييم الشروط في نهاية التكرار

```
Do
        statement(-block)
Loop While condition

Do
        statement(-block)
 Loop Until condition
```

كما يمكن ان تخمن الجمل في جسم الحلقة يتم تنفيذها مرة واحدة على الأقل, لانه لايوجد اختبار عند الدخول في الحلقة,إليك امثلة نموذجية عن استخدام هذه الحلقة Do. . .Loop,افترض ان المتغير MyText يحفظ بعض النص(مثل خاصية Textالنص لأداة صندوق النص TextBox)وتريد ان تحصي الكلمات في النص(سنفترض انه لايوجد عدة فراغات في النص وان ميزة الفراغ تفصل الكلمات المتلاحقة)لايجاد نسخة من الحرف في النص,استخدم الطريقة IndexOfوالتي ستناقش بالتفصيل في الفصل 13 , وهذه الطريقة تقبل معاملين نسبيين: مكان نص البحث والحرف الذي يتم البحث عنه,الحلقة التالية يتم تكرارها طالما انه يوجد فراغات في النص,وفي كل مرة تجد الطريقة IndexOf فراغ أخر في النص، تعود بموقع الفراغ.عندما لا يكون هناك فراغات في النص,تُرجِع الطريقة IndexOf القيمة 1- والتي تشير الى نهاية التكرار(الدوران)كما يظهر هنا:

```
Dim MyText As String = "The quick brown fox jumped over the lazy dog"
      Dim position, words As Integer
      position = 0 : words = 0
      Do While position >= 0
         position = MyText.IndexOf(" ", position + 1)
         words += 1
      Loop
      MsgBox("There are " & words & " words in the text")
```

الحلقة Do. . .Loop يتم تنفيذها بينما دالة الطريقة IndexOf تعود بعدد موجب,والذي يعني انه يوجد فراغات أخرى في النص , المتغير *position* يحفظ موقع رمز الفراغات المتلاحقة في النص,البحث عن الفراغ التالي يبدأ من موقع الفراغ الحالي مع 1 plusإضافة واحد(بالطبع غير ذلك سيبقى البرنامج محتفظا بإيجاد نفس الفراغ الذي في الموقع الأول للفراغ)ومن اجل كل فراغ تم إيجاده يعمل البرنامج على زيادة القيمة للمتغير *words*بمقدار واحد(من احصاء عدد الفراغات في النص)والذي يحفظ العدد الكلي للكلمات عندما ينتهي التكرار.على فكرة يوجد طريقة ابسط لتقسيم النص الى الكلمات المكونة له its constituent مثل الطريقة Split لفئة النص String class . هذا فقط مثال عن الحلقة Do. . .While.,ربما لاحظت المشكلة في مقطع الكود السابق : اني افترضت ان النص يحوي على الأقل كلمة واحدة,عليك إدخال جملة القرار التي تحقق من النص الذي ليس لديه طول(نص بطول صفريzero-length strings)وبالتالي ليس عليك المحاولة في عد كلماته.
تستطيع كتابة الكود لنفس الإجراء مع الكلمة المحجوزة Until.في هذه الحالة عليك الاستمرار في البحث عن الفراغات حتى يصبح المتغير *position* 1- إليك نفس الكود ولكن مع حلقة مختلفة:

```
Dim position As Integer = 0
      Dim words As Integer = 0
      Do Until position = -1
         position = MyText.IndexOf(" ", position + 1)
         words = words + 1
      Loop
  MsgBox("There are " & words & " words in the text")
```

ثالثا- الحلقة (بينما..نهاية بينما)*While...End While*
هذه الحلقة تنفذ بلوك من الجمل طالما الشرط محقق,للحلقة الشكل العام التالي:

```
While condition
        Statement_block
End While
```

اذا كان الشرط condition صحيحا فان جميع العبارات في البلوك سيتم تنفيذها،عندما يصل التنفيذ الى العبارة End While (نهاية بينما)يرجع التحكم الى عبارة While "بينما",والتي تعمل على تقييم الشرط مرة أخرى,فإذا الشرط مازال صحيحا,يتم اعادة العملية.اما اذا كان الشرط غير صحيح (خطأ)يتابع البرنامج مع الجمل التي تلي عبارة End While"نهاية بينما" الحلقة التالية تطلب من المستخدم بيانات عددية, يستطيع المستخدم ان يدخل قيم سالبة ليدل على انه قد أنهى عملية إدخال القيم,وبالتالي ينهي الحلقة.وطالما ان المستخدم يعمل على إدخال قيم عددية موجبة,فان البرنامج يستمر بإضافتها للمتغير totalالكلي:

```
Dim number, total As Double
      number = 0
      While number >= 0
         total = total + number
         number = InputBox("Please enter another value")
      End While
```

عملت على إسناد القيمة 0الى المتغير number قبل ان تبدأ الحلقة لان هذه القيمة ليست قيمة سالبة,ولا تؤثر على المجموع العام.في بعض الأحيان يتم تقيم الشرط الذي يحدد متى تنتهي الحلقة أعلى الحلقة,وفي هذه الحالة,نصرح عن متغير منطقي Boolean ونرجع إليه قيمة صح او خطأ من ضمن جسم الحلقة,إليك الخطوط العريضة لمثل هذه الحلقة:

```
Dim repeatLoop As Boolean
      repeatLoop = True
```

```
    While repeatLoop
      { statements }
        If condition Then
          repeatLoop = True
        Else
          repeattLoop = False
        End If
    End While
```

يمكن ان ترى في بعض الأحيان بلوك حلقات غريبة كتالي:

```
  While True
   { statements }
  End While
```

ومن الشائع تسريع الشرط الصحيح كما يلي:

```
While 1 = 1
```

من الظاهر ان هذه الحلقة لانهائية يجب ان يتم إنهاءها من ضمن جسمها بالعبارةExit While, والتي يتم استدعاءها عندما يصبح شرط ما صح او خطأ,.تنتهي الحلقة التالية عندما يتم مواجهة شرط ما في جسم الحلقة:

```
 While True
     { statements }
        If condition Then Exit While
    { more statements }
  End While
```

Nested Control Structures:تراكيب التحكم المتداخلة

تستطيع ان تضع ,او تدخل تركيب تحكم داخل أخر(مثل بلوك If. . .Then ضمن الحلقةFor. . .Next)فيمكن لتراكيب التحكم في الفيجوال بيسك ان تكون متداخلة وعلى عدة مستويات كما تريد,والمحرر بشكل إلي يترك فراغات مناسبة لتراكيب القرار و الحلقات المتداخلة لجعل البرنامج أسهل من اجل القراءة,.عندما تدخل تراكيب التحكم ضمن بعضها البعض,عليك التأكد من انه يتم فتحها وإغلاقها من نفس التركيب.بعبارة أخرى:لا تستطيع ان تبدأ بالحلقة For. . .Next ضمن عبارة Ifوتغلق الحلقة بعد End If ان تنهي الشرط. يوضح مقطع الكود التالي كيف تدخل عدة عبارات تحكم بالسياق ضمن بعضها البعض(الأقواس المتعرجة تشير الى الجمل النظامية والتي يجب ان يتم استبدالها في مكانها والتي لن يتم ترجمتها بالطبع)

```
For a = 1 To 100
     { statements }
        If a = 99 Then
     { statements }
        End If
        While b < a
    { statements }
          If total <= 0 Then
   { statements }
          End If
        End While
        For c = 1 To a
   { statements }
        Next c
Next a
```

لقد كتبت أسماء متغيرات العداد counterبعد العبارات Next لأجعل من الكود أكثر قابلية للقراءة ومن اجل إيجاد عبارة الإغلاق المناسبة(التاليNext, نهاية اذاEnd If, او نهاية بينما End While).

تبين الجمل التالية تركيب متداخل لحلقة والتي تعمل مسح(استكشاف) لجميع عناصر مصفوفة ذات بعدينtwo-dimensional array :

```
Dim Array2D(6, 4) As Integer
    Dim iRow, iCol As Integer
    For iRow = 0 To Array2D.GetUpperBound(0)
        For iCol = 0 To Array2D.GetUpperBound(1)
          Array2D(iRow, iCol) = iRow * 100 + iCol
          Debug.WriteLine(iRow & ", " & iCol & " = " & Array2D(iRow, iCol) & " ")
        Next iCol
    Next iRow
```

الحلقة الخارجية(التي مع العدادiRow)تتفحص كل صف للمصفوفة.بينما تتفحص الحلقة الداخلية جميع العناصر في الصف المحدد بواسطة عداد الحلقة الخارجية(iRow)في كل عملية دوران،بعد ان تكمل الحلقة الداخلية عملها في الصف الذي حددته لها الحلقة الخارجية,يعود عداد الحلقة الخارجية ليزداد بمقدار واحد,والحلقة الداخلية يتم تنفيذها كاملا على الصف الثاني,فتعمل مسح لجميع عناصره.,يتألف جسم الحلقة من جملتين والتي تسند قيمة ما الى عنصر المصفوفة الحالي ومن ثم تطبعه في نافذة المخرجات,فالعنصر الحالي لدى كل عملية دوران هوArray2D(iRow, iCol) .تستطيع ايضا ان تعمل على مداخلة عبارات متعددة,فالكود التالي يختبر القيمة المعطاة من قبل المستخدم لتحديد فيم اذا كانت موجبة فإذا كانت كذلك فانها تحدد فيم اذا كانت هذه القيمة تتجاوز حد معين.

```
Dim Income As Decimal
    Income = Convert.ToDecimal(InputBox("Enter your income"))
    If Income > 0 Then
        If Income > 12000 Then
          MsgBox("You will pay taxes this year")
        Else
          MsgBox("You won’t pay any taxes this year")
        End If
    Else
        MsgBox("Bummer")
    End If
```

تم مقارنة المتغير"الدخل" Incomeاولا مع صفر,فإذا كان سالبا,فان العبارة Elseللجملة If. . .Then سيتم تنفيذها وستظهر الرسالة(MsgBox("Bummer"), أما اذا كان موجب فانه يتم مقارنته مع القيمة 12000 وبالاعتماد على نتيجة هذه المقارنة فانه سيتم عرض رسالة مختلفة,مقطع الكود المبين هنا لا ينجز أي تحقيقات شاملة ويفترض ان المستخدم لن يعمل على إدخال نص عندما يطلب ا منه لبرنامج ان يكتب دخلهincome.

The Exit Statement: العبارة خروج

تسمح لك العبارة Exit "خروج" الخروج من مقطع الجمل prematurely بشكل سابق لأوانه(قبل إتمام تنفيذ كل الجمل)في تركيب تحكم ما, او من حلقة ما,أو حتى من إجراء ما, افترض ان لديك الحلقة Next. . For والتي تحسب الجذر التربيعي square root لسلسلة من الأعداد, ولأن الجذر التربيعي للاعداد السالبة لا يمكن حسابه(فالطريقة Math.Sqrt ستولد خطأ وقت التنفيذ)

```
For i = 0 To UBound(nArray)
      If nArray(i) < 0 Then
MsgBox("Can't complete calculations" & vbCrLf &"Item " & i.ToString & " is negative! ")
         Exit For
      End If
      nArray(i) = Math.Sqrt(nArray(i))
   Next
```

فإذا تم إيجاد عنصر سالب في هذه الحلقة فان البرنامج يخرج من الحلقة ويستمر في تنفيذ الجمل التالية للجملة . يوجد عبارات خروج مشابهة من اجل الحلقة(Do…loop)(Exit Do)والحلقة While loopوهي:(Exit While) ,وايضا عبارة اختر Selectوهي(Exit Select),وايضا توجد عبارات خروج من اجل الوظائف(الدالات)و الإجراءات الفرعية وهي: (Exit Function and Exit Sub),فإذا كانت الحلقة السابقة جزء من وظيفة فربما تريد إظهار الخطأ والخروج ليس فقط من الحلقة ولكن ايضا من الوظيفة نفسها باستخدام التعبير Exit Function

كتابة واستخدام الإجراءات Writing and Using Procedures

الفكرة في تقسيم التطبيقات الكبيرة الى أقسام اصغر ومقاطع قابلة للترتيب و الإدارة ليست بالجديدة في عالم البرمجة.فالعديد من المهمات البرمجية او ما يشابها يمكن ان يتم إدارتها وترتيبها بالكامل,.فمعالجة الحدث هو فقط مثال واحد عن تقسيم التطبيقات الكبيرة الى مهمات اصغر.فمثلا عندما تكتب كود لحدث النقر, فأنت تركز على الاسم المستعمل,ولكن كيف سيتفاعل البرنامج مع حدث النقر؟ ماذا يحدث عندما يتم نقر الأداة بشكل مزدوج؟او اذا ما تم النقر على اداة أخرى,هذه الأشياء ستجعلك قلقا فيما يخص برمجة الأحداث, فهي permeates تخرق لغة الفيجوال بيسك, او حتى ولو تم كتابة التطبيقات الكبيرة بواسطة أقسام اصغر فهل سيتم التعرف على هذه الأقسام بشكل جيد,و هل من السهولة التحكم بها؟من اجل التساؤلات السابقة كل مهمة يتم إتمامها بواسطة إجراء مستقل والذي يتم اختباره بشكل منفصل عن البقية.

كما ذكرت سابقا, نوعي الإجراءات المدعومة بواسطة الفيجوال بيسك هما : الإجراء الفرعي والوظيفة(البعض يطلق عليها الدالة),فالإجراء الفرعي عادة ينجز أفعالا ولا يعود بأي نتيجة.بينما الوظيفة من الناحية الأخرى, تقوم بعمل بعض الحسابات وتعود بقيمة.و هذا فقط الفرق الوحيد بين الإجراءات الفرعية والوظائف,فكلاهما الإجراء الفرعي والوظائف يمكنها قبول معاملات نسبية:والتي هي القيم التي تمررها للاجراء عندما تستدعيه,وعادة المعاملات النسبية Argumentsهي القيم التي يعمل وفقا لها كود الإجراء.

الإجراءات الجزئية(الفرعية) Subroutines

الإجراءات هي مقطع من العبارات والتي تقوم بمهمة محددة تماما,ومقطع الجمل يتم وضعه ضمن مجموعة عبارات Sub...End Sub, ويمكن استدعاءها بالاسم

الإجراء الجزئي التالي يعرض التاريخ الحالي في صندوق رسالة ويمكن استدعاءه بواسطة اسمه:

```
Sub ShowDate()
   MsgBox(Now().ToShortDateString)
End Sub
```

من الطبيعي أن تكون المهمة المنفذة بواسطة الإجراء الجزئي أن تكون أكثر تعقيدا من المعروضة هنا,ولكن حتى هذا الإجراء الجزئي هو مقطع من كود معزول عن باقي التطبيق.يتم تنفيذ عبارات الإجراء الجزئي وعندما يصل التنفيذ إلى نهاية العبارة End Sub يعود التحكم إلى البرنامج الذي استدعاه. من المكن الخروج من الإجراء بشكل مسبق باستخدام العبارة Exit Sub . جميع المتغيرات التي يتم الإعلان عنها ضمن الإجراء هي متغيرات محلية بالنسبة للإجراء.فعندما سيتم الخروج من الإجراء تتوقف جميع المتغيرات المصرح عنها ضمنه من اجل عملية الخروج.

معظم الإجراءات تقبل ايضا التصرف تبعا للمعاملات النسبية,فالإجراء الجزئي ShowDate() يعرض التاريخ الحالي في صندوق نص,اذا أردت أن يعرض أي تاريخ أخر عليك أن تعالجه بشكل مختلف وتضيف معامل نسبي له:

```
Sub ShowDate(ByVal birthDate As Date)
   MsgBox(birthDate.ToShortDateString)
End Sub
```

"تاريخ الميلاد" birthDateهو متغير يحفظ تاريخ ليتم عرضه,ونوعه typeتاريخ(زمن),الكلمة المحجوزة ByValتعني أن الإجراء الجزئي(الفرعي)يرى نسخة من المتغير وليس المتغير نفسه.ما يعني هذا عمليا أن الإجراء الفرعي لا يستطيع أن يغير قيمة المتغير الممررة إليه بواسطة التطبيق الذي استدعاه.لعرض التاريخ الحالي في صندوق رسالة،عليك استدعاء الإجراء الجزئي كما يلي من ضمن برنامجك:

ShowDate()

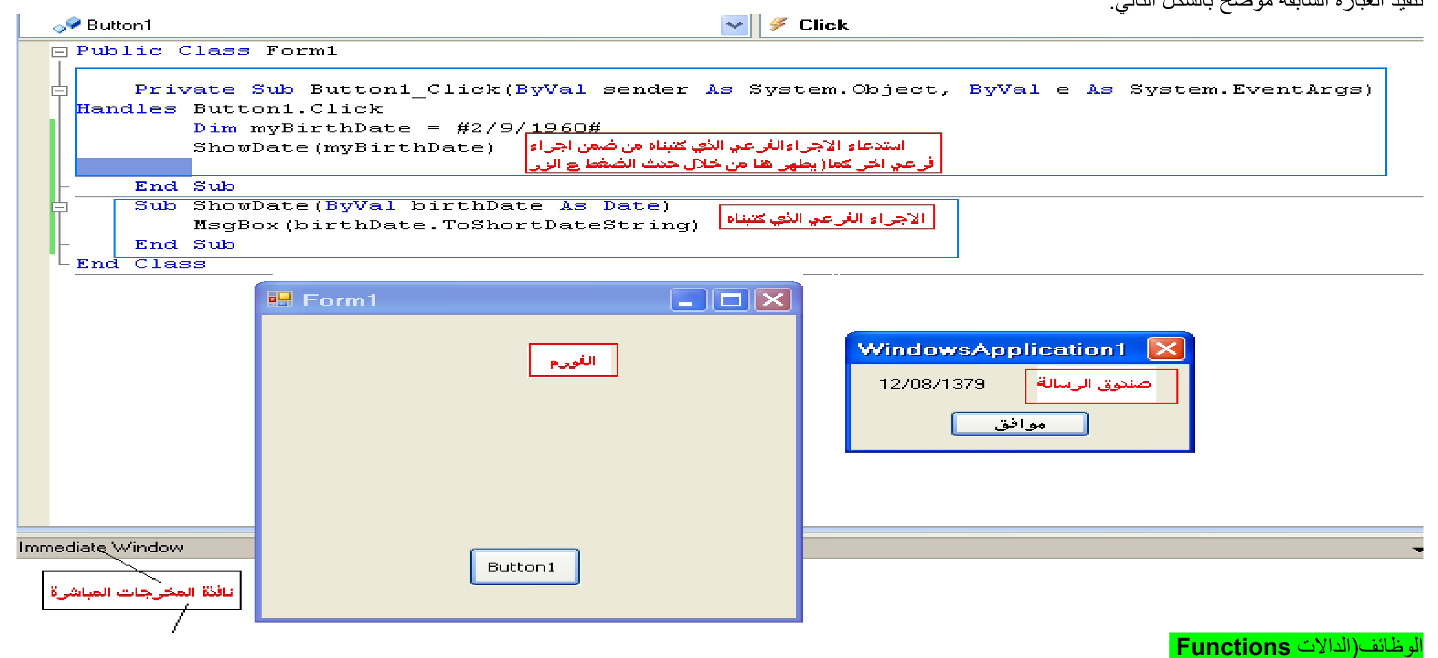
لعرض أي تاريخ أخر بالمعالجة الثانية للإجراء الفرعي. استخدم عبارة مثل التالية:

```
Dim myBirthDate = #2/9/1960#
   ShowDate(myBirthDate)
```

أو تستطيع أن تمرر القيمة ليتم عرضها مباشرة بدون استخدام المتغير كوسيط

```
ShowDate(#2/9/1960#)
```

اذا قررت فيما بعد أن تغير تنسيق التاريخ.يوجد فقط مكان واحد في كودك عليك تحريره: العبارة التي تعرض التاريخ من ضمن الإجراء الفرعي ShowDate() : تنفيذ العبارة السابقة موضح بالشكل التالي:



الوظائف(الدالات) Functions

الدالة مشابهة تماما للإجراء الجزئي, ولكن الوظيفة تعود بنتيجة ما,وبسبب القيم المعادة من قبلها لذا فان الوظائف تشبه المتغيرات من حيث أن لديها أنواع.القيمة التي تمررها من الوظيفة والتي يتم إعادتها إلى الإجراء الذي يستدعيها,تدعى "القيمة المعادة" ونوعها يجب يطابق نوع الوظيفة
تقبل الوظائف معاملات نسبية تماما مثل الإجراءات الفرعية.الجمل التي تصنع وظيفة ما يتم وضعها في مجموعة من جمل"Function. . .End Function " كما هو مبين هنا:

```
Function NextDay() As Date
    Dim theNextDay As Date
    theNextDay = Now.AddDays(1)
    Return theNextDay
End Function
```

Object Browser | Start Page | Form1.vb* | Form1.vb [Design]*

Form1 | NextDay

```
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim myBirthDate = #2/9/1960#
        ShowDate(myBirthDate)
        ShowDate(#2/9/1960#)
    End Sub
    Sub ShowDate(ByVal birthDate As Date)
        MsgBox(birthDate.ToShortDateString)
    End Sub
    Function NextDay() As Date
        Dim theNextDay As Date
        theNextDay = Now.AddDays(1)
        Return theNextDay
    End Function
End Class
```

حدث الضغط على الزر

مجموعة الاجراء الفرعي

مجموعة الوظيفة

فئة الفورم(النموذج)

الكلمة المحجوزة Functionمتبوعة باسم الوظيفة والكلمة المحجوزة Asالتي تحدد نوع الوظيفة،بشكل مشابه تماما للتصريح عن المتغيرات. "إضافة يوم" AddDays هي طريقة لنوع التاريخ,وهي تضيف عدد من الأيام للتاريخ الحالي,الوظيفة ()NextDay تعود بتاريخ يوم الغد بإضافة يوم إلى التاريخ الحالي,الوظيفة NextDay هي وظيفة مخصصة,والتي تستدعي الطريقة AddDaysالمبنية داخليا(المعدة مسبقا)لإتمام الحسابات.
نتيجة الوظيفة يتم اعادتها إلى البرنامج المستدعي لها,من خلال عبارة Return والتي يتم إتباعها بالقيمة التي تريد اعادتها من وظيفتك,هذه القيمة والتي عادة ما تكون متغير يجب أن يكون من نفس نوع الوظيفة,في مثالنا:العبارة المعادة حدث لان تكون أخر عبارة في الوظيفة,ولكن يمكن أن تظهر في أي مكان ومن الممكن ايضا أن تظهر عدة مرات في كود الوظيفة.في المرة الأولى يتم فيها تنفيذ العبارة المعادة ويتم إنهاء الوظيفة,ويعود التحكم للبرنامج المستدعي لها.تستطيع ايضا أن ترجع قيمة ما إلى الإجراء لمستدعيه وذلك بإسناد النتيجة إلى اسم الوظيفة.
الطريقة لتالية بديلة عن كتابة كود الوظيفة ()NextDay:

```
Function NextDay() As Date
    NextDay = Now.AddDays(1)
End Function
```

لاحظ هذه المرة إنني قمت بإسناد نتيجة الحساب إلى اسم الوظيفة مباشرة,ولن استخدم المتغير,هذا الإسناد لاينه الوظيفة مثل عبارة Return"اعد" إنها تُعِد قيمة الوظيفة المعادة,ولكن ستنتهي الوظيفة عندما يتم الوصول إلى العبارة End Function" انهي الوظيفة",أو عندما يتم مصادفة العبارةExit Function" اخرج من الوظيفة". وبشكل مشابه تماما للمتغيرات يجب أن يكون للوظيفة اسم unique مفرد ومميز ضمن مجالها(وهذا الكلام ينطبق على الإجراء الفرعي,بالطبع)اذا صرحت عن وظيفة في form نموذج ما,فان اسم الوظيفة يجب أن يكون مفرد ومميز في الفورم, اذا ما صرحت عن وظيفة ما Publicكعام أو Friend صديق,فان اسمها يجب أن يكون مفرد ومميز في المشروع. للوظائف نفس قواعد المجال scope التي يتم تطبيقها على المتغيرات ويمكن أن يتم ابتدائها بالعديد من نفس الكلمات المحجوزة.في الواقع,تستطيع أن تعدل المجال الافتراضي للوظيفة بواسطة الكلمة المحجوزة"عام" Public أو خاص Private أو محمي Protected أو صديق Friend وصديق ومحمي Protected Friend. بالإضافة,لذلك فان للوظائف أنواع تماما كالمتغيرات,ويتم التصريح عنها بالكلمة المحجوزة As"كـ".على فرض أن الوظيفة ()Count Wordsتعمل على إحصاء عدد الكلمات،والوظيفة ()Count Charsتحصي عدد الحروف في نص ما,فان متوسط طول كلمة ما يمكن أن يتم حسابه كما يلي:

```
Dim longString As String, avgLen As Double
longString = TextBox1.Text
avgLen = CountChars(longString) / CountWords(longString)
```

العبارة التنفيذية الأولى تحصل على النص من TextBox1أداة صندوق النص وتسند هذا النص إلى متغير، والذي تم استخدامه فيما بعد كمعامل نسبي لكلا الوظيفتين,عندما يتم تنفيذ العبارة الثالثة فان الفيجوال بيسك يستدعي أولا الوظيفة CountCharsوالوظيفة CountWords مع المعاملات نسبية محددة ومن ثم يقسم النتائج المعادة.تستطيع استدعاء نفس الطريقة التي تستدعي فيها الإجراء الجزئي,ولكن النتيجة لن يتم تخزينها في أي مكان, مثلا, الوظيفة ()Convert يمكن أن تحول النص في صندوق نص ما إلى الحالة الكبيرة, وتعود بعدد الحروف التي قامت بتحويلها,بشكل طبيعي .ستستدعي هذه الوظيفة كما يلي:

```
nChars = Convert()
```

اذا كنت لا تبالي بالنسبة للقيمة المعادة,فانك لن تحتاج إلا إلى تحديث النص على أداة صندوق النص TextBox, سوف تستدعي الوظيفة ()Convertمع العبارة التالية:

```
Convert()
```

المعاملات النسبية:Arguments

الإجراءات الجزئية والوظائف ليست معزولة بالكامل عن باقي التطبيق. معظم الإجراءات procedures تقبل معاملات نسبية من البرنامج المستدعي.,إعادة تسمية هذا بمعامل نسبي argument:وهو قيمة ما تعمل على تمريرها إلى الإجراء والتي بموجبها يعمل عادة الإجراء., وهكذا كيفية التواصل بين الإجراءات الجزئية والوظائف مع بقية التطبيق.
الإجراءات الجزئية والوظائف يمكن أن تقبل أي عدد من المعاملات النسبية,ويجب أن تزود بقيمة ما لكل معامل نسبي للإجراء عندما يتم استدعاءه.,بعض المعاملات النسبية يمكن أن تكون اختيارية وهذا يعني تستطيع أن تحذفها,سترى كيف تعالج المعاملات النسبية الاختيارية,الوظيفة المخصصة()Min مثلا,تقبل عددين وتعود بالأصغر بينهما:

```
Function Min(ByVal a As Single, ByVal b As Single) As Single
    Min = IIf(a < b, a, b)
End Function
```

()IIf هي وظيفة معدة داخليا(جاهزة) والتي تقيم المعامل النسبي الأول,وهو تعبير منطقي,فإذا كان التعبير صحيح فان الوظيفة تعود بالمعامل النسبي الثاني.أما اذا كان هذا التعبير خطأ تعود الوظيفة ()IIfبالمعامل النسبي الثالث, لاستدعاء الوظيفة المخصصة, استخدم بعض العبارات كما يلي:

```
Dim val1 As Single = 33.001
    Dim val2 As Single = 33.0011
```

```
Dim smallerVal As Single
smallerVal = Min(val1, val2)
Debug.Write("The smaller value is " & smallerVal)
```

اذا نفذت هذه العبارات (ضع هذه الجمل في معالج حدث النقر على الزر)سترى التالي في نافذة الإخراج المباشر Immediate window "القيمة الأصغر هي 33.001 The "smaller value is 33.001" اذا حاولت استدعاء نفس الوظيفة بقيمتين من النوع المزدوج ,بعبارة مثل التالية سترى القيمة 3.33في نافذة المخرجات المباشرة

```
Debug.WriteLine(Min(3.33000000111, 3.33000000222))
```

يعمل المترجم على تحويل كلا القيمتين من المزدوج إلى نوع البيانات المفرد,ويرجع واحد منهما,.فأي منهما؟ليس هناك أي فارق طالما التحويل يتم إلى النوع المفرد,فكلا القيمتين نفس الشيء.الشيء المهم الذي سيحدث اذا حاولت أن تستخدم الوظيفة ()Min مع خيار التدقيق Strict option وهو فعال on,ادخل العبارة Option Strict On "خيار التدقيق فعال" في بداية الملف وقبل كل عبارة.أو ضع خيار التدقيق إلى فعال من صفحة خصائص المشروع كما تعلمت سابقا,ترى أن المترجم يضع خط متعرج تحت العبارة التي تعالج الوظيفة()Min, فالوظيفة ()IIf تقبل متغيران من النوع الكائني Object variables كمعاملات نسبيةarguments, وتعود بأحدهما كنتيجة لها,فخيار التدقيق يمنع المترجم من تحويل الكائن إلى قيمة عددية لاستخدام الوظيفة مع خيار التدقيق ()IIf عليك أن تغير معالجته كما يلي:

```
Function Min(ByVal a As Object, ByVal b As Object) As Object
    Min = IIf(Val(a) < Val(b), a, b)
End Function
```

من الممكن أن تنفذ الوظيفة ()Min التي تستطيع ان تقارن المعاملات لجميع الأنواع(صحيح integers و strings نصي, تاريخ dates, وهكذا)سنعود إلى هذا المثال البسيط فيما بعد في هذا الفصل"إعادة تعريف الوظائف""Overloading Functions.''

آليات تمرير المعاملات النسبية:Argument-Passing Mechanisms

واحد من أهم المواضيع في معالجة إجراءاتك هو موضوع الآلية المستخدمة في تمرير المعاملات النسبية,فالأمثلة حتى الآن قد استخدمت الآلية الافتراضية:تمرير المعاملات النسبية بالقيمة byvalue.الآلية الاخرى في تمرير المعاملات النسبية هي التمرير بالمرجع by reference,على الرغم من أن معظم المبرمجين يستخدمون الآلية الافتراضية, ولكن من الهام أن تعرف الاختلاف بين كلا الآليتين ومتى تستخدم أي منهما.

بالقيمة مقابل بالمرجع By Value versus by Reference

عندما تمرر معامل نسبي ما بالقيمة,فان الإجراء(الإجراء الفرعي او الوظيفة) فقط يرى نسخة من المعامل النسبي,حتى ولو غيرها الإجراء , فالتغيرات غير ثابتة the changes aren't permanent,بكلمة أخرى قيمة المتغير الأصلي الممررة إلى الإجراء لن تتأثر. الفائدة من تمرير المعاملات النسبية بالقيمة هو أن قيم المعامل النسبي معزولة عن الإجراء,وفقط مقطع الكود الذي تم التصريح عنها به يستطيع أن يغير قيمها,وهذه هي الآلية الافتراضية في تمرير المعاملات النسبية في الفيجوال بيسك 2008, في الفيجوال بيسك6 كانت آلية التمرير الافتراضية للمعاملات النسبية بالمرجع.وهذا شيء يجب أن تأخذه في حسبانك,.وخاصة اذا عملت على ترحيل كود فيجوال بيسك6 إلى الفيجوال بيسك2008 .
لتحديد المعاملات النسبية التي سيتم تمريرها بالقيمة.استخدم الكلمة المحجوزة ByVal أمام اسم المعامل النسبي. فاذا ما حذفت الكلمة المحجوزة ByVal,فان المحرر سيعمل على إدخالها بشكل أوتوماتيكي,لأنها الخيار الافتراضي.,
للتصريح على أن المعامل النسبي للوظيفة ()Degrees يتم تمريره بالقيمة. استخدم الكلمة المحجوزة ByVal في تصريح المعامل النسبي كما يلي:

```
Function Degrees(ByVal Celsius As Single) As Single
    Return ((9 / 5) * Celsius + 32)
End Function
```

لترى ما تعمله الكلمة المحجوزة ByVal أضف السطر التي تغير قيمة المعامل النسبي في الوظيفة:

```
Function Degrees(ByVal Celsius as Single) As Single
    Celsius = 0
Return ((9 / 5) * Celsius + 32)
End Function
```

والان استدعي الوظيفة كمايلي:

```
Dim ctemp As Integer
    CTemp = InputBox("Enter temperature in degrees Celsius")
    MsgBox(CTemp.ToString & " degrees Celsius are " & Degrees((CTemp)) & " degrees Fahrenheit")
End Sub
```

32 degrees Celsius are 89.6 degrees Fahrenheit

بدل الكلمة المحجوزة ByVal بالكلمة المحجوزة ByRef في تعريف الوظيفة واستدعي الوظيفة كما يلي:

```
Dim ftemp As Single
    Dim celsius As Single
    Celsius = 32.0
    FTemp = Degrees(Celsius)
MsgBox(Celsius.ToString & " degrees Celsius are " & FTemp & " degrees Fahrenheit")
```

هذه المرة سيعرض البرنامج الرسالة التالية:"0 درجة سلسيوس هي 89.6 درجة فهرنهايت"
0 degrees Celsius are 89.6 degrees Fahrenheit
عندما تم تمرير المعامل النسبي الى الوظيفة,كانت قيمته32 .ولكن الوظيفة غيرت هذه القيمة(غيرت قيمته) وعند الإعادة كانت القيمة 0 .لان المعامل النسبي تم تمريره بالمرجع،فأي تغير يتم عمله في الإجراء يؤثر على المتغير بشكل ثابتpermanently .بالنتيجة عندما حاول البرنامج المستدعي ان يستخدم المتغير كان للمتغير قيمة مختلفة عن تلك المتوقعة.

إرجاع قيم متعددة Returning Multiple Values

اذا أردت كتابة وظيفة تعود بأكثر من نتيجة واحدة,فعلى اغلب الأحوال عليك ان تمرر additional arguments معاملات نسبية إضافية وبالمرجعByRef وتضع قيم هذه المعاملات من ضمن كود الوظيفة.فالوظيفة "()CalculateStatistics الإحصاءات المحسوبة" التي ستراها فيما بعد في هذا المقطع,تحسب الإحصائيات الضرورية لمجموعة بيانات,.فيم مجموعة البيانات يتم تخزينها في مصفوفة تم تمريرها الى الوظيفة بالمرجع,فالوظيفة ()CalculateStatistics يجب ان تعود بقيمتين :وهما المتوسط average والانحراف المعياري standard deviation لمجموعة البيانات, إليك التصريح عن الوظيفة()CalculateStatistics :

```
Function CalculateStatistics(ByRef Data() As Double, ByRef Avg As Double, ByRef StDev_   As Double) As Integer
End Function
```

تعود الوظيفة بـInteger والتي هي عدد القيم في مجموعة البيانات,.والقيمتين الهامتين المحسوبتين من قبل الوظيفة تم اعادتهم في المعاملات النسبية *Avg* و *StDev*

```
Function CalculateStatistics(ByRef Data() As Double, ByRef Avg As Double, ByRef StDev As Double) As Integer
    Dim i As Integer, sum As Double, sumSqr As Double, points As Integer
    points = Data.Length
    For i = 0 To points - 1
      sum = sum + Data(i)
sumSqr = sumSqr + Data(i) ^ 2
    Next
    Avg = sum / points
StDev = System.Math.Sqrt(sumSqr / points - Avg ^ 2)
    Return (points)
End Function
```

لاستدعاء الوظيفة من ضمن كودك, قم بإعداد مصفوفة من النوع المزدوج وصرح عن متغيرين لحفظ المتوسط والانحراف المعياري لمجموعة البيانات:

```
Dim Values(99) As Double
    ' Statements to populate the data set
    Dim average, deviation As Double
    Dim points As Integer
    points = CalculateStatistics(Values, average, deviation)
    Debug.WriteLine(points & " values processed.")
    Debug.WriteLine("The average is " & average & " and")
    Debug.WriteLine("the standard deviation is " & deviation)
```

استخدام المعاملات النسبية بالمرجع هي ابسط طريقة للوظيفة التي ترجع بعدد من القيم,ولكن مهمايكن,الانحراف في الدالة التي استخدمتها يمكن ان يصبح غير مرتب(فوضوي cluttered)وخاصة اذا أردت ان تعود بقيم أكثر.يوجد مشكلة أخرى في هذه التقنية وهي انها غير واضحة بشان المتغيرات النسبية هل يجب ان يتم إسناد قيم له قبل استدعاء الوظيفة.كما سترى باختصار من الممكن بالنسبة للوظائف ان تعود بمصفوفة او تركيب مخصص مع الحقول و بأي عدد من القيم

Passing Objects as Argumentsتمرير الكائنات كمعاملات نسبية

عندما تمرر الكائنات كمعاملات نسبية فانه يتم تمريرها بالمرجع,حتى ولو خصصت لها الكلمة المحجوزة"بالقيمة" يستطيع الإجراء الوصول وتعديل مكونات الكائن الممررة كمعاملات نسبية,وستكون قيمة جديدة مرئية في الإجراء الذي صنع الاستدعاء.مقطع الكود التالي يوضح هذا. فالكائن هو ArrayList قائمة مصفوفة والتي تم تحسينها من مصفوفة ,وقائمة المصفوفة ArrayList تم مناقشتها بالتفصيل لاحقا, ولكن لتتبع هذا المثال كل ما تحتاج معرفته هو ان الطريقة Add"إضافة" تعمل على إضافة بند جديد الى قائمة المصفوفة،وتستطيع ان تصل الى البنود المفردة بواسطة قيمة index الاندكس(الفهرس) بشكل مشابه لعناصر المصفوفة. في معالج حدث الضغط على الزر انشأ نسخة جديدة من كائن قائمة المصفوفة واستدعي الإجراء الفرعي ()PopulateList لملئ القائمة.حتى ولو تم تمرير كائن قائمة مصفوفة الى الإجراء الجزئي بالقيمة فان الإجراء الجزئي سيصل الى بنوده

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Dim aList As New ArrayList()
    PopulateList(aList)
    Debug.WriteLine(aList(0).ToString)
    Debug.WriteLine(aList(1).ToString)
    Debug.WriteLine(aList(2).ToString)
  End Sub
```

نفس الكلام ينطبق على المصفوفة وجميع collections المجمعات الأخرى.حتى ولو خصصت الكلمة المحجوزة بالقيمة,فانه تم التمرير بالمرجع,.يوجد طرق elegant method جيدة أخرى من اجل تعديل مكونات التركيب من ضمن إجراء ومنها معالجة الإجراء كوظيفة تعود بتركيب ،كما سيتم شرحها في مقطع"Functions Returning Structures," " الوظائف التي تعود بتراكيب" فيما بعد في هذا الفصل.

Built-in Functionsالوظائف المعدة داخليا(الجاهزة)

تزود الفيجوال بيسك2008 بالعديد من الوظائف التي تعالج مهمات مشتركة او معقدة,وتستطيع ان تجدها في المستندات(المساعدةMSDN)ستجدهم بالتفصيل في التالي☹(فيجوال استوديو_فيجوال بيسك_المراجع_فرع الوظائف لشجرة المحتويات في مستندات فيجوال استوديو)

Visual Studio_ Visual Basic _ Reference_ Functions branch of the contents tree in the Visual Studio documentation.)

فيوجد وظائف من اجل العمليات الرياضية المشتركة,وظائف لانجاز الحسابات مع التاريخ(وهذه بالحقيقة عمليات معقدة جدا),وظائف التمويلfinancial ,والعديد العديد من الوظائف الأخرى. وعندما تستخدم الوظائف الجاهزة ليس عليك معرفة كيف تعمل داخليا,فقط عليك معرفة كيف تستدعي هذه الوظائف وكيفية استخلاص القيم المعادة.

فالدالة(الوظيفة()Pmt)مثلا تحسب المدفوعات الشهرية على قرض ما,كل ما تحتاج معرفته هو المعاملات النسبية التي يجب ان تمررها الى الوظيفة وكيفية استخلاص النتيجة.الشكل العام للوظيفة ()Pmtهو التالي:حيث ان: MPayهي المدفوع شهريا , Rate:نسبة الفائدة:هي الفائدة الشهرية المرغوبة(الموضوعة) NPer: عدد مرات الدفع(فترة القرض بالشهر). PV القيمة الحالية للقرض(الكمية التي اخذتها من البنك):

MPay = Pmt(Rate, NPer, PV, FV, Due)

أما Due: فهو معامل نسبي اختياري ويحدد متى سيتم دفع المستحق الشهرية(في أول الشهر او في نهاية الشهر)وايضا المعامل FV: هو معامل نسبي اختياري أخر والذي يحدد القيمة المستقبلية للكمية.وهذه المعاملات ليس لها حاجة في حالة القرض,ولكن يمكن ان تساعدك على حساب كمية النقود التي deposit ستودعها في البنك كل شهر لاستكمال accumulate الكمية المطلوبة خلال الفترة المقررة للقرض.(الكمية المعادة بواسطة الوظيفة هي قيمة سالبة لأنها نقود مستحقة عليك(أنت مدين بها owe للبنك وستدفعها في كل شهر حتى يتراكم المبلغ الكلي المستحق عليك خلال فترة القرض المقرر ان تستوفي فيها كامل القرض)

To calculate the monthly payment for a $20,000 loan paid off over a period of six years at a fixed interest rate of 7.25%, you call the Pmt() function, as shown in Listing

لحساب المدفوع شهريا من اجل مثلا قرض كميته $20,000خلال فترة ستة سنوات عند نسبة فائدة ثابتة سنويا بـ7.25% استدعي الوظيفة Pmt كما في الكود التالي:

```
Dim mPay, totalPay As Double
    Dim Duration As Integer = 6 * 12
    Dim Rate As Single = (7.25 / 100) / 12
    Dim Amount As Single = 20000
    mPay = -Pmt(Rate, Duration, Amount)
totalPay = mPay * Duration
    MsgBox("Your monthly payment will be " & mPay.ToString("C") & vbCrLf & "You _ will pay back a total of " &
totalPay.ToString("C"))
```

لاحظ ان نسبة الفائدة تم تقسيمها على 12 لان الوظيفة تطلب الفائدة الشهرية المقصودة وليس الفائدة السنوية. والقيمة المعادة بواسطة الوظيفة Pmt هو mPay المدفوع شهريا للقرض المحدد بالمتغيرات: Duration الفترة والكمية Amount ونسبة الفائدةRate *فإذا وضعت السطور السابقة في معالج حدث الضغط على زر ,شغل المشروع ومن ثم اضغط على الزر ستظهر لك الرسالة التالية في صندوق رسالة:*

Your monthly payment will be $343.39 (مدفوعك شهريا سيكون $343.39)

You will pay back a total of $24,723.80 (ستدفع كامل المبلغ المسترجع بمقدار $24,723.80)

Let's say you want to accumulate $40,000 over the next 15 years by making monthly deposits of equal amounts. To calculate the monthly deposit amount, you must call the Pmt() function, passing 0 as the present value and the target amount as the future value. Replace the statements in the button's Click event handler with the following and run the project

لنقول انك تريد ان تضع نقود في البنك حتى تتراكم وتصبح $40,000 في 15الخمسة عشر السنة القادمة وذلك باستيداع كميات شهرية متساوية خلال هذه 15 سنة.لحساب الكمية التي ستودعها شهريا,عليك استدعاء الوظيفة ()Pmt وتمرر لها0 كقيمة حاليةPV ,والقيمة المقصودة(المطلوبةFV القيمة المستقبلية)بدل الجمل في حدث الضغط على الزر بالتالي ومن ثم جرب المشروع:

```
Dim mPay As Double
    Dim Duration As Integer = 15 * 12
    Dim Rate As Single = (4.0 / 100.0) / 12
    Dim Amount As Single = -40000.0
    mPay = Pmt(Rate, Duration, 0, Amount)
    MsgBox("A monthly deposit of " & mPay.ToString("C") _
                                                  & vbCrLf & "every month will yield $40,000 in 15 years")
```

مايظهره هذا الكود هو انك اذا أردت ان تراكم$40,000 خلال 15 سنة ,على فرض4% معدل فائدة سنوي ثابت ,عليك ان تودع شهريا$162.54,وسوف تضع أنت في البنك تقريبا$30,000 والباقي سيكون الفائدة التي تربحها.

الوظيفة واحدة من ابسط وظائف التمويل والمزودة من قبل إطار العملFramework ,ولكن معظمنا يجد انه من الصعوبة كتابة الكود لهذه الوظيفة لان الحسابات التمويلية شائعة تماما في البرمجة الموجهة للأعمال,فالعديد من الوظائف التي تحتاجها موجودة مسبقا وكل ماتريد معرفته هو كيفية استدعاءها,فإذا كنت مطور تطبيقات تمويلية عليك ان تبحث عن وظائف التمويل

في المستندات.دعنا ننظر الى وظيفة جاهزة أخرى وهي اسم الشهر()MonthName , والتي تقبل كمعاملات نسبية,رقم الشهر وتعود باسم الشهر.هذه الوظيفة ليست بالوظيفة التافهة كما يمكن ان تعتقد لأنها تعود باسم الشهر او لأنها اختصار abbreviation في اللغة للويندوز المحلي(الإعدادات الإقليمية للغة) .الوظيفة MonthName تقبل معاملات نسبية رقم الشهر وقيمة صح/خطأ والتي تحدد فيما اذا سوف تعود باختصار او بالاسم الكامل للشهر,الجمل التالية تعرض اسم الشهر الحالي(كلاهما الاختصار والاسم الكامل)في كل مرة تنفذ فيها هذه الجمل سوف ترى اسم الشهر الحالي في للغة الحالية(لإعدادات اللغة التي لديك في نظام كمبيوترك)

```
Dim mName As String
      mName = MonthName(Now.Month, True)
      MsgBox(mName) ' prints "Jan"
      mName = MonthName(Now.Month, False)
      MsgBox(mName) ' prints "January"
```

وظيفة أخرى مشابهة للسابقة وهي الوظيفة والتي تعود باسم اليوم لأسبوع محدد ,وهذه الوظيفة تقبل معامل نسبي اضافي يحدد اليوم الأول من الأسبوع(راجع مستندات ميكروسوفت لمعلومات اضافية حول الشكل العام للوظيفة)الدور لرئيسي للوظائف هو توسيع functionality الفعالية (الناحية الوظيفية) للغة. العديد من الوظائف التي تقوم إلى حد ما بالعمليات التطبيقية المشتركة أو العامة تم تضمينها في اللغة,ولكنها ليست كافية تقريبا لاحتياجات المطورين أو لجميع أنواع التطبيقات,هذا بالإضافة إلى الوظائف الجاهزة تستطيع أن تكتب وظائف مخصصة لتبسيط تطوير تطبيقات خاصة.كما سيتم شرحه في المقطع التالي..

الوظائف المخصصة :Custom Functions

معظم الكود الذي نكتبه للفورم يكون في وظيفة مخصصة أو في إجراء جزئي تابع لها,وهذا الإجراء(سواء كان الإجراء الجزئي أو الفورم)يمكننا أن نستدعيه من عدة أماكن في التطبيق.والإجراءات الجزئية هي تماما مثل الوظائف ما عدا أنها لا تعود بقيمة,لذا سوف نركز على معالجة الوظائف المخصصة باستثناء أن للوظائف قيمة راجعة فكل شيء نقوله هنا وفي المقطع التالي ينطبق على الإجراءات الجزئية ايضا.

لنظر على مثال بسيط نوعا ما(ولكنه ليس تافه)لوظيفة تعمل شيء ما مفيد في حقيقة الأمر. يتم تعريف (تحديد) الكتب بواسطة "عدد الكتاب القياسي الدولي" وهذا العدد uniqueمفرد ومميز(ISBN):

(ISBN) international standard book number ويتركب هذا العدد من 12 رقم متبوعة برقم اختبار check digit .لحساب رقم الاختبار هذا تضرب كل رقم من الـ12 رقم بثابت,فالرقم الأول يتم ضربه بـ1 والرقم الثاني يتم ضربه بـ3 والرقم الثالث يتم ضربه بـ1 مرة أخرى وهكذا,.وبعد الانتهاء من عملية ضرب الأرقام يتم تقسيم المجموع على 10. لحساب رقم الاختبار للعدد978078212283 ISBN قم بحساب مجموع النواتج التالية:

9 * 1 + 7 * 3 + 8 * 1 + 0 * 3 + 7 * 1 + 8 * 3 +
2 * 1 + 1 * 3 + 2 * 1 + 2 * 3 + 8 * 1 + 3 * 3 = 99

المجموع هو99 وعندما تقسمه على10 فالباقي هو9 ورقم الاختبار هو10-9 أو 1 وبالتالي "العدد القياسي الدولي للكتاب" الكامل هو(9780782122831 : ISBN).الوظيفة ()ISBNCheckDigit المبينة في الكود التالي تقبل ال12رقم للـISBN كمعاملات نسبية وتعود برقم الاختبار المناسبcheck digit

```
Function ISBNCheckDigit(ByVal ISBN As String) As String
      Dim i As Integer, chksum As Integer = 0
      Dim chkDigit As Integer
      Dim factor As Integer = 3
      For i = 0 To 11
         factor = 4 - factor
         chksum += factor * Convert.ToInt16(ISBN.Substring(i, 1))
      Next
      Return (((10 - (chksum Mod 10)) Mod 10)).ToString
   End Function
```

الوظيفةISBNCheckDigit تعود بقيمة نصية لان أرقام ISBNs تم معالجتها كنصوص وليس أعداد,(ما يؤدي إلى أصفار هام في الـISBN ,ولكن كمجوع ليس له معنى ويتم حذفه في القيم العددية).الطريقة Substring لكائن النص تستخرج عدد الحروف من النص الذي يتم تطبيقها عليه.المعامل النسبي الأول هو موضع نص في النص والثاني هو عدد الحروف التي يجب استخراجها.التعبير(ISBN.Substring(i, 1 يستخرج حرف واحد في كل مرة من متغير النص *ISBN*.خلال الدوران الأول للحلقة فانه يستخرج الحرف الأول,وخلال الدوران الثاني فانه يستخرج الحرف الثاني وهكذا.

الحرف المستخرج هو رقم عددي,والذي تم ضربه بقيمة المتغيرfactor وتم إضافة النتيجة إلى المتغيرchksum .هذا المتغير هو مجموع الاختبارchecksum للـISBN بعد أن تم حساب المجموع قمنا بتقسيمه على 10 وأخذنا باقي القسمة(المعامل الأول Mod يعطي باقي عملية القسمة على عشرة)والتي طرحناها من عشرة والمعامل الثانيMod يضع قيمة العشرة إلى صفر. وهذا هو رقم اختبار الـ ISBN وهو ما تعود به الوظيفة.

تستطيع استخدام هذه الوظيفة في تطبيق يحفظ قاعدة بيانات الكتب للتأكد من أن جميع الكتب التي تم إدخالها بـISBN صحيح.وتستطيع استخدامها مع تطبيقات الانترنت والتي تسمح للمستعرض بان يطلب كتبا بواسطة ISBN الخاص بها,نفس الكود سيعمل مع تطبيقين مختلفين ,حتى عندما تمرره إلى مطورين اخزين,فليس على المطورين الذين يستخدمون الوظيفة التي عملتها أن يعرفوا كيف تم حساب رقم الاختبار,عليهم فقط استدعاء هذه الوظيفة واستخلاص نتيجتها, لاختبار الوظيفةISBNCheckDigit, ابدأ مشروع جديد وضع زر على الفورم ومن ثم ادخل العبارات التالية في معالج حدث الضغط على الزر كما يلي:

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
      Debug.WriteLine("The check Digit is " & ISBNCheckDigit("978078212283"))
   End Sub
```

بعد إدخال كود الوظيفةISBNCheckDigit والكود المستدعي لها كما عملنا عندما تم استدعاء الوظيفة من خلال معالج حدث الضغط على الزر,يمكنك بدل أن يتم إظهار النتيجة في نافذة المخرجات أن تضع أداة صندوق نص على الفورم وتمرر خاصية Text النص لأداة صندوق النص إلى الوظيفة لحساب رقم الاختبار.

تمرير المعاملات النسبية والقيم المعادة:Passing Arguments and Returning Values

تعلمت حتى الآن كيف تكتب وتستدعي الإجراءات مع العديد من المعاملات النسبية وكيفية استخلاص القيم المعادة للوظائف واستخدامها في كودك,في هذا المقطع سنغطي العديد من المواضيع المتقدمة على تقنية تمرير المعاملات النسبية وكيفية كتابة الوظائف التي تعود بقيم متعددة,أو مصفوفات من القيم وأنواع بيانات مخصصة:

تمرير عدد غير معروف من المعاملات النسبية:Passing an Unknown Number of Arguments

بشكل عام جميع المعاملات النسبية التي يتوقعها إجراء ما يتم جدولتها في تعريف الإجراء,والبرنامج الذي يستدعي الإجراء يجب أن يزود بالقيم من اجل جميع المعاملات النسبية.في بعض الحالاتOn occasion ومهما تكن,يمكن أن لا تعرف كم عدد المعاملات النسبية التي سيتم تمريرها إلى الإجراء,.فالإجراءات التي تحسب المتوسطات averagesأو بشكل عام، معالجة قيم متعددة تقبل من القليل إلى العديد من المعاملات النسبية والتي عددها غير معروف وقت التصميم,لذا فان الفيجوال بيسك2008 يدعم الكلمة المحجوزة ParamArrayوالتي تسمح لك بتمرير عدد متغير من المعاملات النسبية إلى إجراء ما.دعنا ننظر إلى المثال التالي:افرض انك تريد أن تملأ أداة صندوق قائمةListBox بالعناصر,لإضافة بنود إلى أداة صندوق القائمة,استدعي الطريقة "إضافة Add "للمجمع التابع لعناصرها كما يلي:

```
ListBox1.Items.Add("new item")
```

هذه الجملة تضيف نص "بند جديد"new item إلى أداة صندوق القائمة .اذا اضفت عدة بنود بشكل متلاحق إلى أداة صندوق القائمة من ضمن كودك, تستطيع كتابة إجراء جزئي والذي ينجز هذه المهمة.الاجراء التالي يضيف عدد متغير من المعاملات النسبية إلى أداة صندوق القائمة:

```
Sub AddNamesToList(ByVal ParamArray NamesArray() As Object)
      Dim x As Object
      For Each x In NamesArray
         ListBox1.Items.Add(x)
      Next x
   End Sub
```

هذه المعاملات النسبية للإجراء الجزئي هي مصفوفة مسبوقة بالكلمة المحجوزة ParamArray والتي تحفظ جميع المعاملات parametersالممررة إلى الإجراء الجزئي. اذا كانت مصفوفة المعاملات the parameter arrayتحفظ بنود من نفس النوع,تستطيع أن تصرح عن مصفوفة من نوع معين(نصstring, أعداد صحيحةinteger, وهكذا)لإضافة بنود إلى القائمة استدعي الإجراء الفرعي AddNamesToList كمايلي:

```
AddNamesToList("Robert", "Manny", "Renee", "Charles", "Madonna")
```

subroutine is given by the following expression

اذا أردت معرفة عدد المعاملات النسبية الممررة بشكل فعلي إلى الإجراء, استخدم الخاصية Length لمصفوفة المعاملاتparameter array. فعدد المعاملات النسبية التي تم تمريرها إلى الإجراء الفرعي AddNamesToList تعطى بالتعبير التالي:NamesArray.Length()

الحلقة التالية تدور على جميع عناصر المصفوفة NamesArray وتضيف هذه العناصر إلى القائمة:

```
Dim i As Integer
    For i = 0 To NamesArray.GetUpperBound(0)
      ListBox1.Items.Add(NamesArray(i))
    Next i
```

مصفوفات الفيجوال بيسك وكما علمنا من قبل أن فهرس العنصر الأول هو الصفر والطريقة GetUpperBound تعود فهرس العنصر الأخير في المصفوفة.الإجراءات التي تقبل العديد من المعاملات النسبية تعتمد على ترتيب المعاملات النسبية.لحذف بعضا من المعاملات النسبية, عليك استخدام الفاصلة الموافقة,لنقول انك تريد أن تستدعي بعض الإجراءات وتحدد المعاملات النسبية الأول,الثاني،الثالث والرابع.فيجب استدعاء الإجراء كما يلي:

```
ProcName(arg1, , arg3, arg4)
```

المعاملات النسبية لإجراءات فرعية مشابهة تكون عادة مشابهة في stature (الأولوية),وترتيبها لا يعمل أي اختلاف.

الوظائف التي تحسب المتوسط أو الإحصاءات الأساسية الاخرى لمجموعة من الأعداد أو إجراء فرعي يملأ أداة صندوق قائمة أو أداة صندوق مركب هي مرشحة رئيسية prime candidatesلان تنفذ هذه التقنية.

اذا كان الإجراء يقبل عدد متغير من المعاملات التي تكون غير متساوية الأولوية,عليك أن تأخذ بالتقنية المشروحة في المقطع التالي.اذا كانت الوظيفة تقبل مصفوفة معاملات يجب أن تكون هذه المعاملات هي الأخيرة في القائمة وان لا يكون احد المعاملات الاخرى اختياري.

المعاملات النسبية المحجوزة Named Arguments

تعلمت كيف تكتب إجراءات بمعاملات نسبية اختيارية وكيف تمرر عدد متغير من المعاملات إلى الإجراء.

ومع ذلك القيد الرئيسي لآلية تمرير المعاملات النسبية, هو ترتيب المعاملات النسبية.بشكل افتراضي ,فيجوال بيسك يطابق القيم الممررة إلى إجراء ما بالمعاملات النسبية المصرح عنها بالترتيب(وهذا هو السبب في أن المعاملات النسبية التي رايتها حتى الآن تدعى المعاملات النسبية الموضعية*positional*)

هذا التقييد تم رفعه(ترقيته) بواسطة إمكانية(مقدرة)الفيجوال بيسك2008 لتحديد المعاملات النسبية المسماة(المحجوزة) وتستطيع أن تزود بالمعاملات النسبية في أي ترتيب لان التميز هنا بالاسم وليس بترتيب هذه المعاملات في قائمة المعاملات النسبية للإجراء.افترض انك كتبت وظيفة تتوقع ثلاث معاملات نسبية:الاسم,العنوان,وعنوان الاميل.

```
Function Contact(Name As String, Address As String, EMail As String)
```

عندما تستدعي هذه الوظيفة عليك أن تزود ثلاث نصوص والتي تطابق المعاملات النسبية:الاسم,والعنوان,والايميل,في ذلك الترتيب,مهما يكن يوجد طريقة أكثر أمنا لاستدعاء هذه الطريقة:زود المعاملات النسبية بأي ترتيب بواسطة أسماءها.فبدلا من استدعاء الوظيفة كما يلي:

```
Contact("Peter Evans", "2020 Palm Ave.,Santa Barbara, CA 90000", _
    "PeterEvans@xample.com")
```

تستطيع استدعاءها كمايلي:

```
Contact(Address:="2020 Palm Ave., Santa Barbara, CA 90000", _
       EMail:="PeterEvans@example.com", Name:="Peter Evans")
```

المعامل= يسند القيم للمعاملات المسماة لان المعاملات النسبية تم تمريرها بالاسم فتستطيع أن تزود بها و بأي ترتيب تريد.لتختبر هذه التقنية ادخل التصريح التالي للوظيفة في كود الفورم:

```
Function Contact(ByVal Name As String, ByVal Address As String, ByVal EMail As String) As String
    Debug.WriteLine(Name)
    Debug.WriteLine(Address)
    Debug.WriteLine(EMail)
    Return ("OK")
  End Function
```

ومن ثم استدعي الوظيفة من ضمن حدث الضغط على الزر بالعبارة التالية:

```
Debug.WriteLine(Contact(Address:="2020 Palm Ave., Santa Barbara, CA 90000", _
                        Name:="Peter Evans", EMail:="PeterEvans@example.com"))
```

سترى التالي في نافذة المخرجات المباشرة:

```
Peter Evans
2020 Palm Ave., Santa Barbara, CA 90000
PeterEvans@example.com
OK
```

تعرف الوظيفة أي من القيم المطابقة إلى المعامل النسبي وتستطيع أن تعالجهم بنفس الطريقة التي تعالج المعاملات النسبية الموضعية.لاحظ أن تعريف الوظيفة هو نفسه فيما كنت تستدعيها بالمعاملات النسبية الموضعية أو الاسمية,الاختلاف هو في كيفية استدعاء الوظيفة وليس في كيفية الإعلان عنها.فالمعاملات النسبية الاسمية تجعل الكود أكثر أمنا وأسهل للقراءة,ولكن وبسبب أنها تحتاج إلى الكثير من الكتابة,فمعظم المبرمجين لا يستخدمون هذه الطريقة ,بالإضافة إلى ذلك ,عندما يكون المساعد الفوري فعال IntelliSense is on تستطيع أن ترى تعريف الوظيفة عندما تدخل المعاملات النسبية(أثناء كتابة الكود يظهر المساعد الفوري تعريف المتغيرات,الوظائف.....)وهذا م يقلل من swapping مبادلة متغيران بالخطأ.

أنواع أخرى لقيم الوظائف المعادة:More Types of Function Return Values

الوظائف غير مقتصرة على ارجاع أنواع بيانات بسيطة مثل الأعداد الصحيحة او النصوص,من الممكن ان تعود بأنواع بيانات مخصصة وحتى مصفوفات,.إمكانية الوظائف على ارجاع جميع أنواع البيانات تجعلها مرنة جدا ويمكن كتابة الكود لها ببساطة ,لذلك فإننا سوف نستكشفها بالتفصيل في المقطع التالي,.استخدام أنواع بيانات معقدة مثل التراكيب والمصفوفات تسمح لك بكتابة وظائف تعود بقيم متعددة.

الوظائف التي ترجع بتراكيب Functions Returning Structures

افترض انك تريد وظيفة تعود بمدخرات زبون ما و أرصدة حسابه الجارية,حتى الآن,تعلمت انك تستطيع ان تعود بقيمتين او أكثر من وظيفة ما,بتزويد المعاملات النسبية بواسطة الكلمة المحجوزة ByRef.الطريقة الأكثر روعة هي إنشاء نوع بيانات مخصصة(تراكيب) وكتابة وظيفة ما تعود بمتغير من هذا النوع.إليك مثال بسيط عن الوظائف التي تعود بنوع بيانات مخصص,وهذا المثال يضع الخطوط العريضة للخطوات التي يجب ان تعيدها في كل مرة تريد فيها ان تنشئ وظيفة تعود بأنواع بيانات مخصصة:

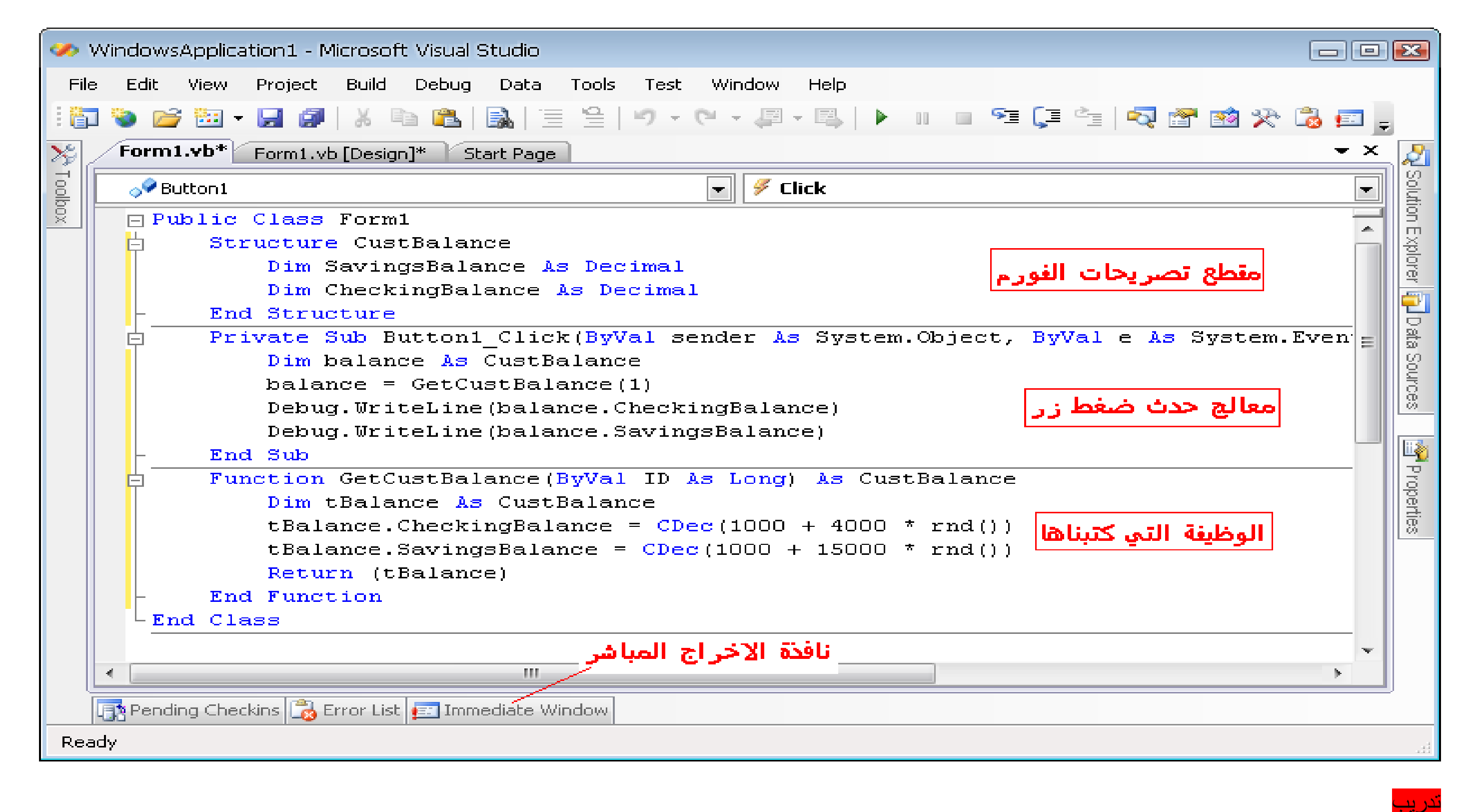
1- أنشئ مشروع جديد وادخل التصريح عن نوع بيانات مخصص في مقطع التصريحات للفورم:

```
Structure CustBalance
    Dim SavingsBalance As Decimal
    Dim CheckingBalance As Decimal
  End Structure
```

2- زود بالوظيفة التي تعود بقيمة من نوع مخصص.في جسم الوظيفة,عليك ان تعلن عن متغير من النوع المعاد بواسطة الوظيفة وتسند القيم المناسبة لحقولها.الوظيفة التالية تسند قيم عشوائية الى حقول"تفحص الرصيد *CheckingBalance* " و"الرصيد المخزن*SavingsBalance* " ومن ثم تعمل على إسناد المتغير الى اسم الوظيفة كما هو مبين☹(ضع هذه الوظيفة خارج أي إجراء)

```
Function GetCustBalance(ByVal ID As Long) As CustBalance
    Dim tBalance As CustBalance
    tBalance.CheckingBalance = CDec(1000 + 4000 * rnd())
    tBalance.SavingsBalance = CDec(1000 + 15000 * rnd())
    Return (tBalance)
  End Function
```

3- ضع اداة زر على الفورم والذي منه ستعمل على استدعاء الوظيفة.صرح عن متغير من نفس النوع واسند له القيمة المعادة بواسطة الوظيفة,المثال التالي يطبع المدخرات الرصيد الجاري في النافذة المخرجات المباشرة:



تدريب

انشأ مشروع جديد وسمه الأنواع وضع عليه الادوات المبينة في الشكل(6 اداة عنوان و6صندوق نص وزرين)وسمهم كما هو مبين

Form1

الشركة

التاريخ

العنوان

المدينة

البلد

الرصيد

السابق اللاحق

```
Public Class Form1
    Structure Customer
        Dim Company As String
        Dim Manager As String
        Dim Address As String التصريح عن تركيب(نوع بيانات مخصص)في مقطع التصريحات العام للفورم
        Dim City As String
        Dim Country As String
        Dim CustomerSince As Date
        Dim Balance As Decimal
    End Structure
    Private Customers(8) As Customer
    Private cust As Customer التصريح عن متغيرات عامة على مستوى الفورم
    Private currentIndex As Integer
    Function CountCustomers() As Integer
        Return (Customers.Length) إنشاء وظيفة تحصي الزبائن
    End Function
    Function GetCustomer(ByVal idx As Integer) As Customer
        Return (Customers(idx)) انشاءوظيفة أخرى تعود بفهرس الزبون
    End Function
    Sub ShowCustomer(ByVal idx As Integer) إجراء فرعي لاسناد القيم الى صناديق النص
        Dim aCustomer As Customer
        aCustomer = GetCustomer(idx)
        txtcompany.Text = aCustomer.Company اسناد قيمة الشركة الى خاصيةالنص لصندوق نص الشركة
        txtsince.Text = aCustomer.CustomerSince.ToShortDateString اسناد قيمة تاريخ الايداع الى نص صندوق نص التاريخ

        txtadd.Text = aCustomer.Address اسناد قيمة العنوان الى خاصية النص لصندوق نص العنوان
```

```
        txtcity.Text = aCustomer.City إسناد قيمة المدينة الى خاصيةالنص لصندوق نص المدينة
        txtcountry.Text = aCustomer.Country إسناد قيمة البلد الى خاصيةالنص صندوق البلد
        txtbalance.Text = aCustomer.Balance.ToString("#,###.00")إسناد قيمة الرصيد الى خاصيةالنص لصندوق نص الرصيد
    End Sub
 حدث تحميل الفورم واضافة عناصر جديد وتعبئة نصوص اوات النص التي على الفورم
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Dim cust As Customer
        cust.Company = "Bottom-Dollar Markets"
        cust.Manager = "Elizabeth Lincoln"
        cust.Address = "23 Tsawassen Blvd."
        cust.City = "Tsawassen"
        cust.Country = "Canada"
        cust.CustomerSince = #10/20/1996#
        cust.Balance = 33500
        Customers(0) = cust
        cust = New Customer()
        cust.Company = "Drachenblut Delikatessen"
        cust.Manager = "Sven Ottlieb"
        cust.Address = "Walserweg 21"
        cust.City = "Aachen"
        cust.Country = "Germany"
        cust.CustomerSince = #1/2/1994#
        cust.Balance = 2400
        Customers(1) = cust
        cust = New Customer()
        cust.Company = "Furia Bacalhau e Frutos do Mar"
        cust.Manager = "Lino Rodriguez "
        cust.Address = "Jardim das rosas n. 32"
        cust.City = "Lisboa"
        cust.Country = "Portugal"
        cust.CustomerSince = #12/22/1998#
        cust.Balance = 300
        Customers(2) = cust
        cust = New Customer()
        cust.Company = "Great Lakes Food Market"
        cust.Manager = "Howard Snyder"
        cust.Address = "2732 Baker Blvd."
        cust.City = "Eugene, OR"
        cust.Country = "USA"
        cust.CustomerSince = #1/3/1998#
        cust.Balance = 6500
        Customers(3) = cust
        cust = New Customer()
        cust.Company = "QUICK-Stop"
        cust.Manager = "Horst Kloss"
        cust.Address = "Taucherstraكe 10"
        cust.City = "Cunewalde"
        cust.Country = "Germany"
        cust.CustomerSince = #1/1/1989#
        cust.Balance = 23400
        Customers(4) = cust
        cust = New Customer()
        cust.Company = "The Cracker Box"
        cust.Manager = "Liu Wong"
        cust.Address = "55 Grizzly Peak Rd."
        cust.City = "Butte"
        cust.Country = "USA"
        cust.CustomerSince = #1/1/1999#
        cust.Balance = 23400
        Customers(5) = cust
        cust = New Customer()
        cust.Company = "White Clover Markets"
        cust.Manager = "Karl Jablonski"
        cust.Address = "305 - 14th Ave. S."
        cust.City = "Seattle, WA"
        cust.Country = "USA"
        cust.CustomerSince = #5/11/1994#
        cust.Balance = 12000
        Customers(6) = cust
        cust = New Customer()
        cust.Company = "Wilman Kala"
        cust.Manager = "Matti Karttunen"
        cust.Address = "Keskuskatu 45"
        cust.City = "Helsinki"
```

```vb
        cust.Country = "Finland"
        cust.CustomerSince = #1/3/2000#
        cust.Balance = 2500
        Customers(7) = cust
        btnnext.PerformClick()لاتقلق بشأنه سيتم شرحه فيمابعد
    End Sub
    Private Sub btnnext_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnnext.Click
        If currentIndex = CountCustomers() - 1 Then currentIndex = 0
        Dim aCustomer As Customer
        aCustomer = GetCustomer(currentIndex)
        ShowCustomer(currentIndex)
        currentIndex = currentIndex + 1
    End Sub
    Private Sub btnprev_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnprev.Click
        If currentIndex = 0 Then currentIndex = CountCustomers() - 1
        Dim aCustomer As Customer
        aCustomer = GetCustomer(currentIndex)
        ShowCustomer(currentIndex)
        currentIndex = currentIndex - 1
    End Sub
End Class
```

في كل مرة تضغط فيها على زر(اللاحق)فان حقل السجل التالي يتم عرضه في أدوات صناديق النصوص الموافقة على الفورم,وعندما يتم استنفاذ جميع الصفوف يلتف البرنامج عائدا الى الصف(السجل الأول).يتألف المشروع من نموذج واحد ويستخدم نوع بيانات مخصص يتم تنفيذه مع التركيب المكتوب في مقطع التصاريح العامة للفورم,وهذا التركيب يجب ان يظهر في كود الفورم وخارج أي إجراء.

المصفوفة *Customers* تحفظ البيانات لـ9 زبائن والمتغير cust تم استخدامه كمتغير مؤقت لتخزين بيانات الزبون الحالي,والمتغير *currentIndex هو فهرس العنصر الحالي للمصفوفة,.حقول المصفوفة مع بيانات الزبون والمتغير currentIndex* تم إسنادها بشكل أولي الى القيمة0 .

معالج حدث ضغط الزر لزر التالي يستدعي الوظيفة()GetCustomer مع قيمة الفهرس(الذي هو ترتيب الزبون الحالي) لاستخلاص بيانات الزبون التالي,وعرض حقول الزبون على أدوات صندوق النص من على الفورم بالإجراء الفرعي ()ShowCustomer.ومن ثم فانه يزيد قيمة المتغير*currentIndex ليشير الى فهرس الزبون الحالي.*

ملاحظة:لقد قمنا بتغير خاصية التسمية البرمجية (name)للأدوات كما هو موضح أسماءها في الكود من خلال نافذة الخصائص(فمثلا زر اللاحق خاصية name= btnnext وزر السابق name= btnprev وهكذا بالنسبة لصناديق النص)

الوظائف التي تعود بمصفوفات:Functions Returning Arrays

بالاضافة الى العودة بأنواع البيانات المخصصة , تستطيع وظائف الفيجوال بيسك2008 ايضا ان تعود بمصفوفات, وهذه إمكانية هامة تسمح لك بكتابة الوظائف ليس فقط التي تعود بقيم متعددة ولكن ايضا بعدد غير معروف من القيم.سنكتب في هذا المقطع وظيفة الإحصاء()Statistics وهي مشابهة لوظيفة الإحصاء الحسابي()CalculateStatistics التي رايتها سابقا في هذا الفصل,الوظيفة "احصاء"()Statistics تعود بالتعداد في مصفوفة ما,وأكثر من ذلك فهي تعود ليس فقط بالمتوسط والانحراف المعياري ولكن ايضا بالقيم الصغرى والعظمى في مجموعة البيانات, يوجد طريقة واحدة للتصريح عن وظيفة تحسب كل الإحصاءات وهي التالية:

```vb
Function Statistics(ByRef DataArray() As Double) As Double()
```

هذه الوظيفة تقبل مصفوفة مع قيم نوع البيانات وتعود بمصفوفة من النوع المزدوج.لتنفيذ وظيفة تعود بمصفوفة عليك ان تعمل التالي:

1- تحديد النوع للقيمة المعادة من المصفوفة و إضافة زوج من الأقواس بعد اسم النوع.لا تحدد طول المصفوفة التي يجب ان ترجع بها هنا,سيتم التصريح عن المصفوفة بشكل شكلي(رسمي) في الوظيفة.
2- في كود الوظيفة,صرح عن مصفوفة من نفس النوع وحدد طولها.فإذا كانت الوظيفة ستعود بأربع قيم, استخدم التصريح التالي: `Dim Results(3) As Double` المصفوفة "Results" النتائج" والتي ستستخدم لتخزين النتائج يجب ان تكون من نفس نوع الذي في الوظيفة ولكن اسمها يمكن ان يكون أي شيء.
3- لإرجاع Resultsالمصفوفة, ببساطة استخدمها كمعامل نسبي لعبارة الإرجاع:`Return(Results)`
4- في الإجراء المستدعي, عليك ان تعلن عن مصفوفة من نفس النوع بدون أبعاد:

```vb
Dim Statistics() As Double
```

5- أخيرا عليك ان تستدعي الوظيفة وتسند قيمتها المعادة الى هذه المصفوفة:

```vb
Stats() = Statistics(DataSet())
```

هنا *DataSet* هي مصفوفة مع القيم التي إحصاءاتها الأساسية سوف يتم حسابها بواسطة الوظيفة()Statistics. ومن ثم فان كودك يستطيع استخلاص كل عنصر للمصفوفة بقيمة الفهرس وكالعادة.

اعادة التعريف للوظائف:Overloading Functions

توجد حالات يكون فيها من الواجب ان تعمل نفس الوظيفة على نوع بيانات مختلف او عدد مختلف من المعاملات النسبية, في الماضي كان عليك ان تكتب وظيفة مختلفة,بأسماء مختلفة ومعاملات نسبية مختلفة,لتلاءم(تكيف accommodate) نفس المتطلبات. قدم إطار العملFramework مبدأ اعادة تعريف الوظيفة,والذي يعني انه بإمكانك ان تحوز على تنفيذات متعددة لنفس الوظيفة,وكل منها مع مجموعة مختلفة من المعاملات النسبية ومن الممكن ايضا قيم معادة مختلفة.ولكن جميع الوظائف المعاد تعريفها تتشارك بنفس الاسم,لأقدم لك هذا المبدأ باستكشاف واحدة من العديد من الوظائف المعاد قيادتها والتي تأتي مع إطار عمل نت NET Framework..الطريقة التالية لفئة System.Random تعود بقيمة صحيحة من–2,147,483,648 الى2,147,483,647(هذا هو مجال القيم التي يمكن تمثيلها بواسطة نوع البيانات Integerالصحيحة)سوف نكون ايضا قادرين على توليد أعداد عشوائية في مجال محدد لقيم صحيحة.لمناقشة(emulate)كش مات(في لعبة الشطرنج)نحتاج الى قيم عشوائية في المجال من(1 الى 6)بينما من اجل لعبة النرد(roulette game القمار)نحتاج الى قيمة عشوائية في المجال من 0 الى 36 تستطيع ان تحدد الحد الأعلى للرقم العشوائي بواسطة معامل نسبي من النوع الصحيح ويكون اختياري.العبارة التالية سوف تعود بقيمة صحيحة عشوائية في المجال من 0 الى 99 :`randomInt = rnd.Next(100)`

تستطيع ايضا ان تحدد كلا الحدين الأعلى والأدنى لمجال العدد العشوائي.العبارة التالية ستعود بقيمة صحيحة عشوائية في المجال من 1000 الى 1999

```vb
randomInt = Rnd.Next(1000, 2000)
```

نفس الطريقة تسلك سلوك مختلف بالاعتماد على المعاملات النسبية التي نزودها نحن,فسلوك الطريقة يعتمد على إما نوع المعاملات النسبية او عددها او كلاهما,ولا يوجد وظيفة وحيدة تبدل سلوكها بالاعتماد على المعاملات النسبية فيوجد العديد من التنفيذات المختلفة لنفس الوظيفة كلما وجد معاملات نسبية مصاحبة لها.جميع هذه الوظائف تتشارك في نفس الاسم لذا فانها تظهر للمستخدم كوظيفة وحيدة ولكن متعددة الواجهات,هذه الوظيفة تم اعادة تعريفها,وسترى كيف يتم تنفيذها في المقطع التالي.فإذا ما كانت المساعدة الفورية تعمل (لم تقم بتعطيلها)فحالما تكتب قوس مفتوح بعد اسم الوظيفة او اسم الطريقة,سترى صندوق اصفر مع الشكل العام للوظيفة او الطريقة.وستعرف ان تلك الوظيفة او الطريقة تم اعادة تعريفها عندما يحتوي هذا الصندوق على عدد و سهمين,كل عدد يوافق نمط مختلف لاعادة التعريف,وتستطيع التحرك الى النمط المعاد تعريفه التالي او السابق وذلك بالضغط على السهمين الصغيرين او بالضغط على مفاتيح الأسهم من لوحة المفاتيح .

لنعود الى الوظيفة ()Min التي تعاملنا معها سابقا في هذا الفصل,التنفيذ الأولي للوظيفة ()Min كمايلي:

```vb
Function Min(ByVal a As Double, ByVal b As Double) As Double
    Min = IIf(a < b, a, b)
End Function
```

بقبولها قيم مزدوجة كمعاملات نسبية هذه الوظيفة تستطيع ان تعامل جميع أنواع القيم العددية,ينجز الفيجوال بيسك2008 التحويلات الواسعة بشكل آلي(يستطيع ان يحول الأعداد الصحيحة والعشرية الى النوع المزدوج)

لذا فان هذه السمة تجعل الوظيفة تعمل مع جميع البيانات العددية,ولكن ماذا بشان النصوص؟فإذا حاولت ان تستدعي الوظيفة مع نصين كمعاملان نسبيان,ستحصل على استثناء(خطأ):الوظيفة Min() لا تستطيع معالجة النصوص

لكتابة وظيفة تستطيع ان تعالج القيم النصية والعددية,في الحقيقة عليك كتابة وظيفتين Min() .جميع الوظائف Min() يجب ان تسبق بالكلمة المحجوزة Overloads, الجمل التالية تبين الاختلاف في تنفيذ نفس الوظيفة:

```
Overloads Function Min(ByVal a As String, ByVal b As String) As String
    Min = Convert.ToString(IIf(a < b, a, b))
  End Function
```

نحتاج الى نموذج اعادة تعريف ثالث لنفس الوظيفة لمقارنة التاريخ.اذا استدعيت الوظيفة Min()ومررت كمعاملات نسبية تاريخين,كما في العبارة التالية,فان الوظيفة ستقارن هذين التاريخين كنصوص وتعود بالتاريخ الأول غير صحيح: Debug.WriteLine(Min(#1/1/2009#, #3/4/2008#)) حتى هذه العبارة غير صحيحةعندما يكون خيار التدقيق فعال Strict option is onولذلك فمن الواضح انك تحتاج الى نموذج اعادة تعريف أخر للوظيفة تقبل تاريخين كمعاملين نسبيين لها كما مبين هنا:

```
Overloads Function Min(ByVal a As Date, ByVal b As Date) As Date
    Min = IIf(a < b, a, b)
  End Function
```

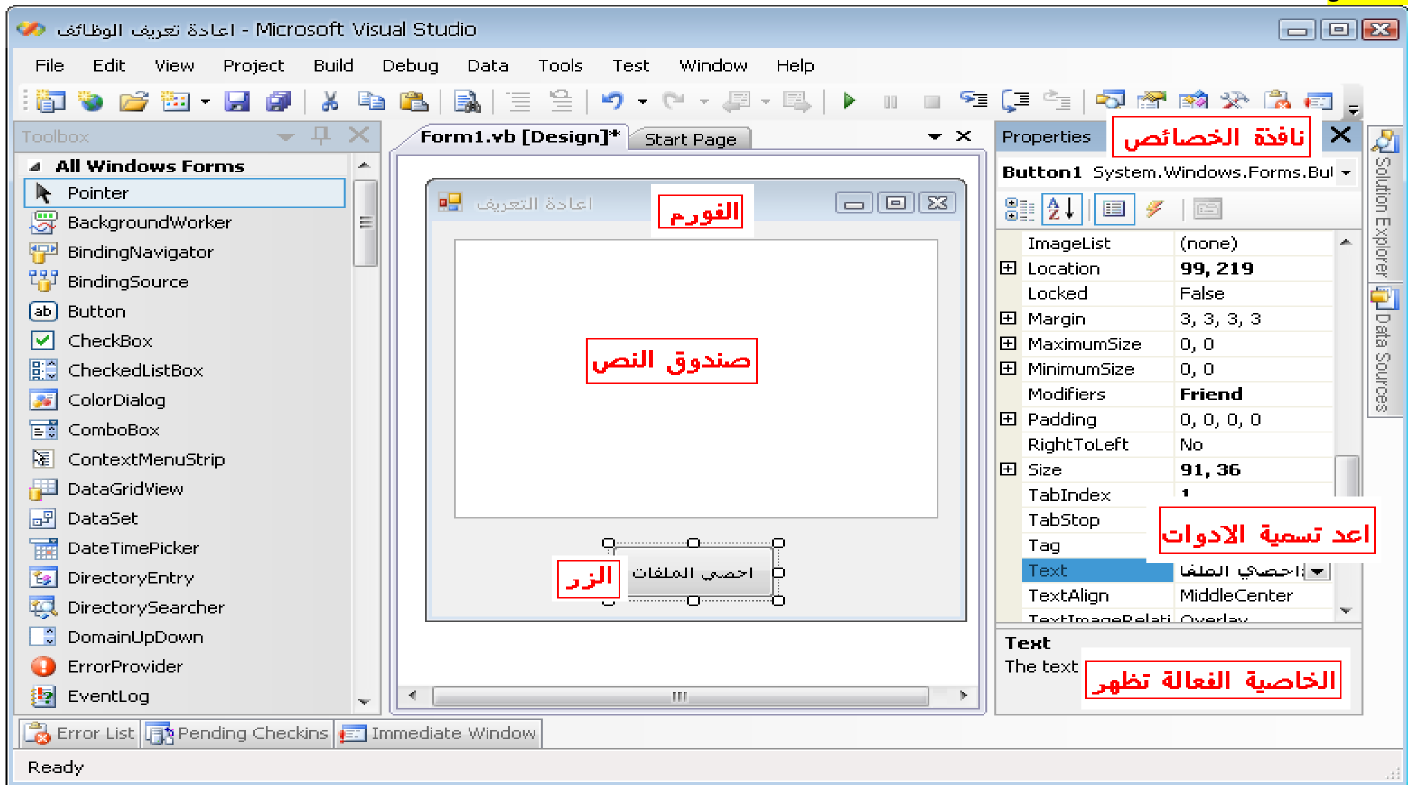
اذا استدعيت الوظيفة مع التاريخين#1/1/2009#و#3/4/2008# ستعود الوظيفة بالتاريخ الثاني,والذي هو بالترتيب الزمني chronologicallyاصغر من الأول.,لنذهب الى وظيفة أكثر تعقيدا,والتي تجعلنا نستخدم بعض المواضيع التي تم مناقشتها لاحقا في الكتاب,الوظيفة"عدد الملفات()CountFiles "تحصي عدد الملفات في المجلد الذي يقابل معيار محدد,وهذا المعيار يمكن ان يكون حجم الملفات,او نوعها,او تاريخ إنشاءها,.تستطيع ان تأتي بأي جمع لهذه المعايير,ولكن التالي هو الجمع الأكثر فائدة(هذه الوظائف التي سأستخدمها,ولكن تستطيع ان تنشئ جموع أكثر او تقدم معايير جديدة خاصة بك).أسماء المعاملات النسبية تصف نفسها,لذلك لا احتاج لشرح ما يفعل كل نموذج للوظيفة()CountFiles

```
CountFiles(ByVal minSize As Integer, ByVal maxSize As Integer) As Integer
CountFiles(ByVal fromDate As Date, ByVal toDate As Date) As Integer
CountFiles(ByVal type As String) As Integer
CountFiles(ByVal minSize As Integer, ByVal maxSize As Integer,ByVal type As String) As Integer
CountFiles(ByVal fromDate As Date, ByVal toDate As Date,ByVal type As String) AsInteger
```

القائمة التالية تبين تنفيذ نماذج اعادة التعريف هذه للوظيفة()CountFiles (سنعمل فيما يلي مشروع اعادة التعريف)ولكن بما أننا لم نناقش العمليات على الملفات بعد,فجميع الكود في جسم الوظيفة سيكون جديد عليك لذا لا تقلق بشان الاكواد الجديدة ستناقش في دراسة العمليات على الملفات,من اجل المنفعة العامة للقارئين الذين لا يعرفون عمليات الملفات, فقد ضمنت عبارات تطبع في نافذة المخرجات نوع الملف الذي تم إحصاءه بواسطة كل وظيفة.ولكن إليك المشروع كاملا:

تدريب

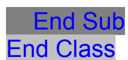
أنشئ مشروع جديد وضع على الفورم زر وصندوق نص اعد تسمية الفورم من خاصية النص لها text الى "اعادة التعريف للوظائف" وسم الزر ايضا من خاصية النص للزر"أحصي الملفات" من نافذة الخصائص كما هو مبين في الشكل المرافق.ملاحظة:لتستطيع ان توسع صندوق النص اجعل خاصية متعددة الأسطر (صح) لصندوق النص multiline=true من نافذة الخصائص



والان اضغط ضغط مزدوج على الزر وادخل الكود التالي كما هو مبين في القائمة التالية:

```
Public Class Form1
    Dim folderName As String = "c:\windows\System32"
    Overloads Function CountFiles(ByVal minSize As Integer, _
                ByVal maxSize As Integer) As Integer
        Debug.WriteLine("You've requested the files between " & minSize & _
                " and " & maxSize & " bytes")
        Dim files() As String
        files = System.IO.Directory.GetFiles(folderName)
        Dim i, fileCount As Integer
        For i = 0 To files.GetUpperBound(0)
          Dim FI As New System.IO.FileInfo(files(i))
          If FI.Length >= minSize And FI.Length <= maxSize Then
            fileCount = fileCount + 1
          End If
        Next
        Return (fileCount)
```

```vb
End Function
Overloads Function CountFiles(ByVal fromDate As Date, _
                              ByVal toDate As Date) As Integer
    Debug.WriteLine("You've requested the count of files created from " & _
             fromDate & " to " & toDate)
    Dim files() As String
    files = System.IO.Directory.GetFiles(folderName)
    Dim i, fileCount As Integer
    For i = 0 To files.GetUpperBound(0)
        Dim FI As New System.IO.FileInfo(files(i))
        If FI.CreationTime.Date >= fromDate And _
           FI.CreationTime.Date <= toDate Then
          fileCount = fileCount + 1
        End If
    Next
    Return (fileCount)
End Function
Overloads Function CountFiles(ByVal type As String) As Integer
    Debug.WriteLine("You've requested the " & type & " files")
    Dim files() As String
    files = System.IO.Directory.GetFiles(folderName)
    Dim i, fileCount As Integer
    For i = 0 To files.GetUpperBound(0)
        Dim FI As New System.IO.FileInfo(files(i))
        If FI.Extension = type Then
          fileCount = fileCount + 1
        End If
    Next
    Return (fileCount)
End Function
Overloads Function CountFiles(ByVal minSize As Integer, _
        ByVal maxSize As Integer, ByVal type As String) As Integer
    Debug.WriteLine("You've requested the " & type & _
            " files between " & minSize & " and " & _
            maxSize & " bytes")
    Dim files() As String
    files = System.IO.Directory.GetFiles(folderName)
    Dim i, fileCount As Integer
    For i = 0 To files.GetUpperBound(0)
        Dim FI As New System.IO.FileInfo(files(i))
        If FI.Length >= minSize And _
           FI.Length <= maxSize And _
           FI.Extension = type Then
          fileCount = fileCount + 1
        End If
    Next
    Return (fileCount)
End Function
Overloads Function CountFiles(ByVal fromDate As Date, _
         ByVal toDate As Date, ByVal type As String) As Integer
    Debug.WriteLine("You've requested the " & type & _
            " files created from " & fromDate & " to " & toDate)
    Dim files() As String
    files = System.IO.Directory.GetFiles(folderName)
    Dim i, fileCount As Integer
    For i = 0 To files.GetUpperBound(0)
        Dim FI As New System.IO.FileInfo(files(i))
        If FI.CreationTime.Date >= fromDate And _
           FI.CreationTime.Date <= toDate And FI.Extension = type Then
          fileCount = fileCount + 1
        End If
    Next
    Return (fileCount)
End Function
Private Sub bttnCount_Click(ByVal sender As System.Object, ByVal e As   System.EventArgs) Handles bttnCount.Click
    TextBox1.AppendText(CountFiles(1000, 100000) & _
          " files with size between 1KB and 100KB" & vbCrLf)
    TextBox1.AppendText(CountFiles(#1/1/2001#, #12/31/2001#) & _
          " files created in 2003" & vbCrLf)
    TextBox1.AppendText(CountFiles(".DLL") & " BMP files" & vbCrLf)
    TextBox1.AppendText(CountFiles(1000, 1000000, ".EXE") & _
          " EXE files between 1 and 100 KB" & vbCrLf)
    TextBox1.AppendText(CountFiles(#1/1/2004#, #12/31/2007#, ".EXE") & _
          " EXE files created from 2004 to 2007")
```



End Class

اذا كنت لا تعرف الكائنات " Directory الدليل:مساحة محدد على القرص والملف File" ركز على العبارات التي تطبع في نافذة الإخراج المباشر Immediate window وتجاهل العبارات التي تحسب بشكل عملي الملفات,تستطيع العودة لهذا المثال وتفهم عباراته فيما بعد عندما ندرس "التمكن من الوصول الى الملفات والمجلدات" كما ترى جميع الوظائف المعاد قيادتها لها نفس الاسم والتركيب وتختلف فقط في كيفية اختيارها للملفات لتعمل على إحصاءها.نفذ المشروع وراقب نافذة الإخراج المباشر التي ستطبع العبارات التالية كما في الشكل الموضح هنا:

You've requested the files between 1000 and 100000 bytes لقد طلبت الملفات التي بين 1000 و100000 بايت
You've requested the count of files created from 1/1/2001 to لقد طلبت الملفات التي تم انشاءها من تاريخ 1/1/2001 الى تاريخ 12/31/2001
You've requested the .DLL files لقد طلبت الملفات من النوع
You've requested the .EXE files between 1000 and 1000000 bytes(ملفات تنفيذية بالحجم المبين)
You've requested the .EXE files created from 1/1/2004 to 12/31/2007(ملفات تنفيذية مع التاريخ من كذا الى كذا)

اعادة تعريف الوظائف (Running) - Microsoft Visual Studio

File Edit View Pro

اعادة التعريف

1406 files with size between 1KB and 100KB
675 files created in 2003
44 BMP files
5 EXE files between 1 and 100 KB
1 EXE files created from 2004 to 2007

اضغط الزر

احصي الملفات

Form1.vb Form1.v
Button1

```
Overload          Integer, _
  Debu            s String) As
                  type & _
                  e & " and " &
  Dim
  file            rName)
  Dim
  For
                  (i))
```

Solution Explorer
اعادة تعريف الوظائف
My Project
Form1.vb
Solution Explorer Data Sources
Properties
Button1_Click Attributes

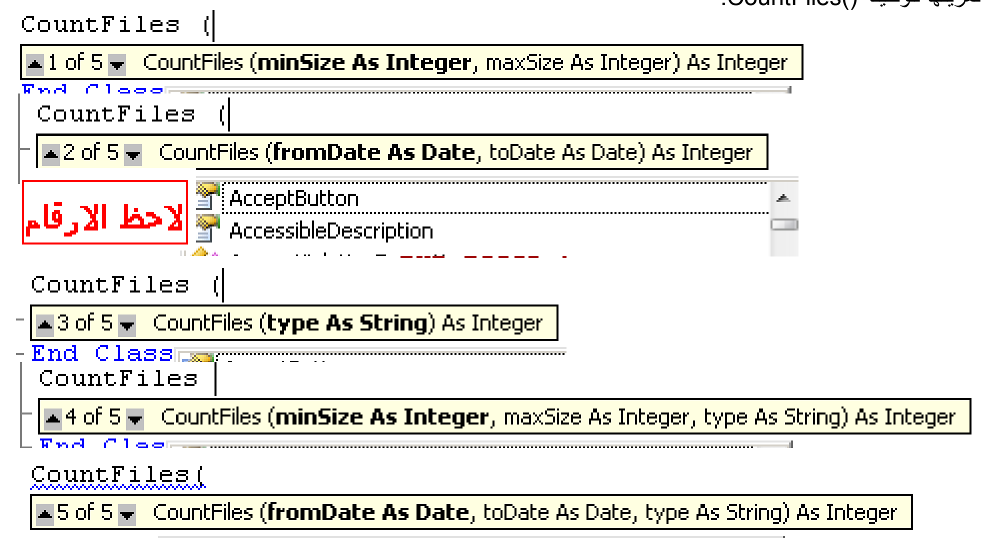
Immediate Window

```
You've requested the files between 1000 and 100000 bytes
You've requested the count of files created from 1/1/2001 to 12/31/2001
You've requested the .DLL files
You've requested the .EXE files between 1000 and 1000000 bytes
You've requested the .EXE files created from 1/1/2004 to 12/31/2007
```

لاحظ نافذة المخرجات المباشرة بعد ان تضغط ع الزر

Immediate Window Error List
Pending Checkins
Ready Ln 92 Col 1 Ch 1 INS

يستدعي الزر النماذج المعاد قيادته المتنوعة للوظيفة واحدة بعد الأخرى ويطبع النتائج ع اداة صندوق النص على الفورم,اعادة تعريف الوظائف يتم استخدمها بكثرة في كامل اللغة,ويوجد نسبيا القليل من الوظائف(او الطرق)لم يتم اعادة تعريفها.في كل مرة تدخل اسم الوظيفة متبوعا بقوس مفتوح فان قائمة بمعاملاته تظهر في قائمة منسدلة تحوي المعاملات النسبية للوظيفة.فإذا كانت هذه الوظيفة قد تم اعادة تعريفها سترى عدد في مقدمة قائمة المعاملات النسبية هذا العدد هو ترتيب نماذج اعادة التعريف للوظيفة ويتم إتباعه بالمعاملات النسبية للنموذج المحدد من الوظيفة.الشكل التالي يريك جميع النماذج المعاد تعريفها للوظيفة CountFiles():



الفصل الرابع

تصميم واجهة المستخدم التصويرية(الرسومية) والبرمجة المقادة بالأحداث GUI Design and Event-Driven Programming

أصبحت تعرف الآن كيف تصمم واجهة المستخدم الرسومية graphical user interfaces (GUI), وكيف تستخدم عبارات الفيجوال بيسك لبرمجة الأحداث لأدوات متنوعة.وتعلمت ايضا كيف تكتب الوظائف والإجراءات الجزئية وكيف تستدعي الوظائف المعدة داخليا (الجاهزة) في الفيجوال بيسك,. في هذا الفصل ستعمل على تصميم العديد من تطبيقات ويندوز,في هذه المرة ستكون تطبيقات عملية مع واجهات أكثر فعالية,وكود يعمل شيء ما بعملية أكثر مما قد رايته سابقا,ففي هذا الفصل ستتعلم مايلي:

ا- تصميم واجهة المستخدم الرسومية
ب- برمجة الأحداث
ج- كتابة تطبيقات قوية robust مع معالج الخطأ

تصميم تطبيقات ويندوز يمر بمرحلتين متميزتين:تصميم واجهة التطبيق وكتابة الكود للتطبيق. يتم انجاز تصميم الواجهة المرئية للتطبيق بواسطة أدوات مرئية ويتألف هذا التصميم من إنشاء الفورم مع العناصر المتعلقة بها relevant وهذه العناصر هي بلوكات بناء تطبيق ويندوز(حجر الأساس)لبناء تطبيقات ويندوز وتسمى الأدوات(او المتحكماتcontrols)

الادوات الممكنة التي تظهر في صندوق الادوات وهي نفسها العناصر التي تستخدم بواسطة جميع تطبيقات ويندوز,بالإضافة الى كونها وافرة بشكل مرئي, تتضمن الادوات الكثير من الوظيفية(الفعالية)، فأداة صندوق النص TextBoxتستطيع ان تعالج النصوص من خلال إمكاناتها الخاصة ،بدون أي جهد برمجي من جهتك, وأداة الصندوق المركب ComboBox تنسدل بقائمة بنودها عندما ينقر المستخدمين زر السهم وتعرض البند المختار في صندوق تحريرها الخاص. بشكل عام التخصص الوظيفي للأدوات معد (مجهز) للأدوات بشكل تصميمي,لذا فان جميع التطبيقات تحافظ على مظهر ثابت.

تملي dictates الواجهة كيفية تفاعل المستخدمون مع تطبيقك .لتطلب من المستخدم ان يدخل نص او بيانات عددية ,استخدم اداة صندوق النصTextBox . عندما يصدف لان تحدد واحد او أكثر من عدة خيارات فلديك العديد من الاختيارات,فتستطيع ان تستخدم اداة الصندوق المركب ComboBoxوالتي منها يستطيع المستخدمون ان يختاروا خيار ما, او تستخدم عدة أدوات لصندوق الاختيار CheckBoxعلى الفورم بحيث يستطيع المستخدمون ان يضعوا علامة صح(يختار) او يزيلوا هذه العلامة(لا يختار) ,اذا أردت ان تعرض عدد قليل من الخيارات المقتصرة على التبادل المشترك(شيء او عكسه) ضع عدد من أدوات زر التحويل RadioButtonعلى الفورم ففي كل مرة يختار المستخدمون خيار ما فان الاختيار السابق يتم إزالته.لتبدأ الأفعال ضع واحد او أكثر من اداة الزر Buttonعلى الفورم.

بناء برنامج حاسب القروض Building a Loan Calculator

تزودك الفيجوال بيسك بالتابع(الوظيفة) التي تقوم بإتمام العديد من أنواع الحسابات المالية,وتحتاج فقط الى سطر واحد من الكود لحساب المدفوع شهريا من خلال كمية القرض,ومدة القرض ونسبة الفائدة,حيث ان تصميم واجهة المستخدم يأخذ عمل أكثر من كتابة الكود بغض النظر عن اللغة المستخدمة في كتابة الكود,لذا عليك إجراء العمليات التالية لبرمجة التطبيق:

1- تقرير ما سيعمله التطبيق وكيفية الاستجابة للمستخدم.
2- تصميم واجهة المستخدم للتطبيق طبقا لمتطلبات الخطوة السابقة.
3- كتابة الكود الفعلي لمعالجة الأحداث التي تريد معالجتها خلف الواجهة.

فهم كيفية عمل تطبيق حاسب القروض Understanding How the Loan Calculator Application Works

بالرجوع الى الخطوة الأولى للخطوط الرئيسية,فأنت قررت ان المستخدم سيكون قادر على تحديد كمية القرض ونسبة الفائدة وفترة القرض بالشهور،لذلك عليك ان تزود التطبيق بثلاث صناديق نصوص يستطيع ان يدخل المستخدم فيها القيم السابقة الذكر ،بالإضافة الى معامل أخر يؤثر على المدفوع شهريا وهو فيم اذا كان المدفوع سيتم مسبقا في أول كل شهر أم في أخره,لذلك عليك ان تزود المستخدم بطريقة يحدد من خلالها هل سيتم الدفع مبكرا(في اليوم الأول من الشهر) او متأخر(في أخر يوم من الشهر),فالنوع الأكثر مناسبة للتحكم في عملية الإدخال هذه هي (نعم او لا)أي (أول الشهر اذا كانت نعم أواخر الشهر اذا كانت لا)(او صح /خطأ) لذا نحن بحاجة الى اداة صندوق اختيارCheckBox, هذه الأداة هي عقدة مفصلية(أي يتم تبديلها من خلال النقر لوضع علامة الصح داخل الصندوق و النقر مرة أخرى لإزالة علامة الصح من داخل صندوق الاختيار) فلن يقوم المستخدم بإدخال أية بيانات في هذه الأداة (وهذا يعني انك لا تريد ان يبادر المستخدم الى إدخال أي خطأ في هذه الأداة) وهذه ابسط طريقة لتحديد القيم لحالة لا تتقبل إلا واحد من الاحتمالين.
يريك الشكل التالي واجهة المستخدم التي تطابق مواصفات تطبيقنا,وهذه هي الفورم الرئيسية لمشروع حاسب القروض

حساب المدفوع الشهري لقرض ما

كمية القرض 25000

الفائدة السنوية 14.5

الفترة(بالشهر) 48

الدفع في اول يوم من الشهر

المتوجب دفعه كل شهر

خروج حساب المدفوع الشهري

يعمل المستخدم على إدخال جميع المعلومات على الفورم ومن ثم يضغط على زر المدفوع شهريا لحساب المدفوع شهريا.وبالتالي سيعمل البرنامج على حساب المدفوع الشهري ويعرضه في أسفل اداة من أدوات صندوق النص ,يحدث جميع العمل في الإجراء الجزئي التابع لحدث النقر على الزر.

لحساب المدفوع الشهري نعمل على استدعاء الوظيفة الجاهزة(() Pmt) والتي لها الشكل العام التالي:

Monthly Payment = Pmt(InterestRate, Periods, Amount, FutureValue, Due)

عملنا على مناقشة هذه الوظيفة في الفصول السابقة ولكن سأعمل على شرح بعض الضروريات هنا فإذا كانت نسبة الفائدة السنوية مثلا 14.5 % ستكون الفائدة الشهرية 12/0.145 , أما فترة القرض فيعبر عنها المعامل النسبي Periodsبالشهر,أما الكمية Amount كمية القرض, FutureValue: قيمة القرض عند نهاية الفترة والتي ستكون في حالة القرض صفر,بينما ستكون موجبة في حالة الاستثمارinvestment . أما المعامل الأخير Dueيحدد متى سيتم الدفع.قيمة هذا المعامل Dueيمكن ان تكون واحدة من الثوابت DueDate.BegOfPeriod وDueDate.EndOfPeriod, وهذان الثابتان هما جاهزان من اللغة(تم إعدادهما بشكل مسبق) وحتى انك تستطيع ان تستخدمهما بدون ان تعرف قيمة كل منهما بالضبط.

تصميم واجهة المستخدم Designing the User Interface

بعد ان عرفت كيف تحسب المتوجب دفعه كل شهر,تستطيع تصميم واجهة المستخدم,ولعمل هذا ,ابدأ مشروع جديد وسمه "حاسب القروض" وأعد تسمية الفورم الى"حساب المدفوع الشهري لقرض ما" , حدد الخط وحجمه الذي تراه مناسبا للفورم لان هذا الخط سيؤثر على باقي الادوات التي ستقوم بوضعها على الفورم.اجعل خاصية RightToLeft من اليمين الى اليسار(نعم) واجعل ايضا RightToLeftLayout (صح) وذلك من اجل الكتابة باللغة العربية وليصح تخطيط النموذج لديك من اليمين إلى اليسار

سنستخدم في الكتاب نوع الخط فيردانا 10 نقاط10-point Verdana font لتغير الخط اختر الفورم(بالنقر على أي مكان ع الفورم) واذهب الى نافذة الخصائص واختر الخاصيةFont لفتح صندوق حوار الخط واختر الخط الذي تريده.أنصحك باستخدام أما(سيغو Seago او Verdana) لأنهما الخطان المخصصان للعرض على الشاشات.لتصميم الواجهة المبينة في الشكل السابق اتبع الخطوات التالية:

1- ضع أربع أدوات labels من اداة العنوان على الفورم وذلك لكتابة العناوين فيها (يتم ذلك في خاصية Textالنص لكل منهم)

الاسمName	خاصية النص Text
Label1	كمية القرض
Label2	الفائدة السنوية
Label3	الفترة(شهر)
Label4	المتوجب دفعه كل شهر

لست بحاجة لتغير الاسم الافتراضي لأدوات العناوين السابقة على الفورم ما نحتاجه هو العنوان فقط من خلال خاصية النص ,فلن تعمل على برمجة هذه الادوات أبدا فهي مجرد عناوين فقط.

2- ضع اداة صندوق نص مقابل كل اداة عنوان وضع خاصية الاسم Name وخاصية النصText الى القيم التالية .حيث ان خاصية الاسم Name ستستخدم في الكود لذا اختر اسم يعبر عنها أما خاصية النص فضع الأرقام الموضحة في الشكل السابق إليك التوضيح وكما يلي:

الاسم Nameقبل التغير	الاسم Nameبعد التغير	خاصية النص Text
TextBox1	txtAmount	25000
TextBox2	txtRate	14.5
TextBox3	txtDuration	48
TextBox4	txtPayment	اتركه فارغ

3- أداة صندوق النص الأخيرة تركناها فارغة لأننا سنعمل على إظهار (المتوجب دفعه في كل شهر) في هذه الأداة, فليس من المفترض ان يدخل المستخدم في هذه الأداة أي بيانات, لذا عليك ان تجعلها للقراءة فقط من خلال خاصية ReadOnly ضع هذه الخاصية الى صح(True) لمنع المستخدم من الكتابة فيها.تستطيع التحكم في قيمة هذه الخاصية من خلال الكود,ولكن المستخدم لن يستطيع الكتابة فيها(يمكننا بالمقابل ان نستخدم اداة عنوان بدلا عن اداة صندوق النص)

4- ضع اداة صندوق اختيارCheckBox1 على الفورم ومن خاصية النص **Text** وكما هو موضح في الشكل السابق ضع النص " الدفع في أول يوم من الشهر",وبشكل افتراضي سيكون العنوان لها CheckBox1. وفي نافذة الخصائص ايضا ضع خاصية الاسم " Name" الى "**chkPayEarly** "
5- ضع اداة زر على الفورم وغير خاصية **Text** النص كما هو موضح في الشكل السابق الى"حساب المدفوع الشهري" وغير خاصية **Name**الاسم الى "bttnShowPayment "
6- أخيرا ضع زر أخر على الفورم وضع خاصية النص له الى "خروج" وخاصية الاسم الى "bttnExit " كما هو موضح في الشكل السابق.

برمجة تطبيق حاسب القروض Programming the Loan Application

الآن وبعد ان عملت واجهة المستخدم , شغل التطبيق وشاهد سلوكه ,ادخل عدد من القيم في صناديق النصوص واضغط على زر حساب المدفوع الشهري ,ماذا تلاحظ ؟ تجد ان البرنامج لا يقوم بحساب أي شيء ,لذا أوقف تشغيل البرنامج واضغط مزدوج على زر "حساب المدفوع الشهري "وادخل فيه الكود التالي:

```
Dim Payment As Double
    Dim LoanIRate As Double
    Dim LoanDuration As Integer
    Dim LoanAmount As Integer
    LoanAmount = Convert.ToInt32(txtAmount.Text)
    LoanIRate = 0.01 * Convert.ToDecimal(txtRate.Text) / 12
    LoanDuration = Convert.ToInt32(txtDuration.Text)
    Dim payEarly As DueDate
    If chkPayEarly.Checked Then
      payEarly = DueDate.BegOfPeriod
    Else
      payEarly = DueDate.EndOfPeriod
    End If
    Payment = Pmt(LoanIRate, LoanDuration, -LoanAmount, 0, payEarly)
    txtPayment.Text = Payment.ToString("#.00")
```

نافذة محرر الكود ستبدو موافقة للشكل التالي:

```
Form1.vb*   Form1.vb [Design]*   Start Page
ShowPayment                                   Click

Public Class Form1

    Private Sub ShowPayment_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ShowPayment.Click
        Dim Payment As Double
        Dim LoanIRate As Double
        Dim LoanDuration As Integer
        Dim LoanAmount As Integer

        LoanAmount = Convert.ToInt32(txtAmount.Text)
        LoanIRate = 0.01 * Convert.ToDecimal(txtRate.Text) / 12
        LoanDuration = Convert.ToInt32(txtDuration.Text)
        Dim payEarly As DueDate
        If chkPayEarly.Checked Then
            payEarly = DueDate.BegOfPeriod
        Else
            payEarly = DueDate.EndOfPeriod
        End If
        Payment = Pmt(LoanIRate, LoanDuration, -LoanAmount, 0, payEarly)
        txtPayment.Text = Payment.ToString("#.00")
    End Sub
End Class
```

اذا أردت ان تقسم الأسطر الطويلة يدويا عليك ان تدخل في المكان الذي تريد ان تقسم السطر "خط منخفض" او من قائمة "تحرير Edit " اختار خيارات متقدمة "Advanced " ومنه اختار "التفاف النص Word Wrap" في الكود السابق قمنا اولا بالتصريح عن المتغيرات التي سنخزن فيها القيم التي نريد فالمتغيرPayment (يخزن قيمة من النوع المزدوج) وهذا المتغير سنخزن فيه "المتوجب دفعه كل شهر" بعد ان يتم حسابه عند الضغط على زر "حساب المدفوع الشهري" وبعد حساب المدفوع في كل شهر نعمل على إظهاره في نص صندوق نص "المتوجب دفعه في كل شهر" بعد ان نعمل على تحويله الى نص بالتنسيق المبين في السطر الأخير من الكود أي السطر التالي:

```
txtPayment.Text = Payment.ToString("#.00")
```

أما التغير الثاني LoanIRate فإننا سنخزن فيه نسبة الفائدة الشهرية وذلك كما ترى قمنا بإسناد نص صندوق نص الفائدة السنوية بعد تحويلها الى عدد عشري و عملنا على تحويل الفائدة السنوية الى معدل الفائدة في كل شهر وذلك بتقسيمها على عدد شهور السنة وهي 12 شهر وضربها بـ0.01 لتحويلها الى نسبة مئوية.

أما المتغيرLoanDuration فقد عملنا على تخزين مدة القرض بالشهور فيه واسندنا له نص صندوق نص الفترة(شهر) بعد ان قمنا بتحويلها الى النوع عدد صحيح,

أما المتغير الأخير فهو LoanAmountفقد ادخلناه لتخزين كمية القرض فيهقد قمنا بإسناد قيمة نص صندوق نص كمية القرض الى هذا المتغير بعد ان قمنا بتحويلها الى قيمة عددية صحيحة,وهو من نوع عدد صحيح.

أما السطر التالي من الكود Dim payEarly As DueDate فانه يصرح عن متغير من نوع "تابع زمني DueDate " ونوع المعامل النسبي هذا يحدد فيما اذا سيتم الدفع مبكرا في أول الشهر أم في نهاية الشهر حيث ان المعامل النسبي الأخير للوظيفة ()Pmt هو متغير من هذا النوع كما أوضحنا سابقا,وحيث ان DueDate هو عداد يحوي مكونين وهما BegOfPeriod و EndOfPeriod(بداية الشهر ونهاية الشهر على الترتيب)

```
Dim payEarly As DueDate
    If chkPayEarly.Checked Then
        payEarly = DueDate.BegOfPeriod
    Else
        payEarly = DueDate.EndOfPeriod
    End If
```

كما ترى بعد ان قمنا بالتصريح عن متغير من النوع DueDate فقد ادخلنا تركيب شرطي , لاختبار صندوق الاختيار فإذا ما تم اختياره(وضع علامة الصح داخل صندوقه) سيتم تنفيذ الكود payEarly = DueDate.BegOfPeriod أي ان الدفع سيتم مقدما في بداية الشهر, وإلا فان الدفع سيتم متأخر في نهاية الشهر عند اذ سينفذ الكود التالي في حالة عدم وضع علامة الصح في صندوق الاختيار payEarly = DueDate.EndOfPeriod
الآن نفذ البرنامج واضغط على زر حساب المدفوع الشهري ,غير القيم في صناديق النصوص (ولكن لا تدخل الآن إلا قيم رقمية لأننا لم نعالج حتى الآن الخطأ الذي سينتج عن إدخال قيم نصية في صناديق النص), اضغط على صندوق الاختيار لوضع علامة الصح فيه وراقب التغيرات ومن ثم ازل هذه العلامة , تدرب قليلا على البرنامج وراجع كيفية تنفيذ الكود وعندما تنتهي ,اغلق البرنامج للعودة الى بيئة التطوير مرة أخرى (لا تغلق البرنامج من خلال زر الخروج فإننا لم نعمل على برمجته حتى الآن)

ملاحظة:
اذا كنت لا تعرف شيئا حول الوظيفة الجاهزة Pmt,وكيفية حساب المدفوعات الشهرية؟ يوجد Code snippets قصاصة كود جاهزة ,وكما شرحنا سابقا قصاصة الكود اضغط في مكان ما من الكود باليمين واختار القائمة المنسدلة "إدخال قصاصة كود Insert Snippet" واضغط مزدوج على "fundamentalsالأساسيات" لترى قائمة أخرى من البنود, اضغط مزدوج من هذه البنود على "مجلد الرياضيات Math" ومن ثم اختار القصاصة "Calculate a Monthly Payment   on a Loan" سيتم إدخال الكود التالي:

```
Dim futureValue As Double = 0
    Dim payment1 As Double
    payment1 = Pmt(0.05 / 12, 36, -1000, futureValue, DueDate.EndOfPeriod)
```

توضح هذه القصاصة استخدام الوظيفة Pmt كل ما عليك عمله هو تبديل قيم المعاملات في هذه الوظيفة بالبيانات من الأداة المناسبة التي قمت بوضعها على الفورم.
وإذا كنت لاتعرف ايضا كيف تستخدم المعاملات النسبية للوظيفة , ضع المؤشر فوق كل معامل نسبي وسترى الشرح لكل معامل نسبي كما هو مبين هنا:

```
payment = Pmt(0.05 / 12, 36, -1000, futureValue, DueDate.EndOfPeriod)
```
Replace with a Double for the annual interest rate (e.g. 0.05 for 5%).

```
payment = Pmt(0.05 / 12, 36, -1000, futureValue, DueDate.EndOfPeriod)
```
Replace with code that returns a Double for the loan amount.

```
payment = Pmt(0.05 / 12, 36, -1000, futureValue, DueDate.EndOfPeriod)
```
Replace with code that returns a Double for the total number of monthly payments.

لكن كودالمشروع البسيط "حاسب القروض" نوعا ما مختلف واطول مما قدمته هنا,حيث انه يمكن للمستخدم ان يدخل جميع أنواع البيانات في الفورم فإذا ما ادخل نص ما سيؤدي الى حدوث خطأ ,لذا سترى في المقطع القادم كيف تعمل على التحقق من البيانات المدخلة من قبل المستخدم, واصطياد الاخطاء ومعالجتها

التحقق من البيانات :Validating the Data

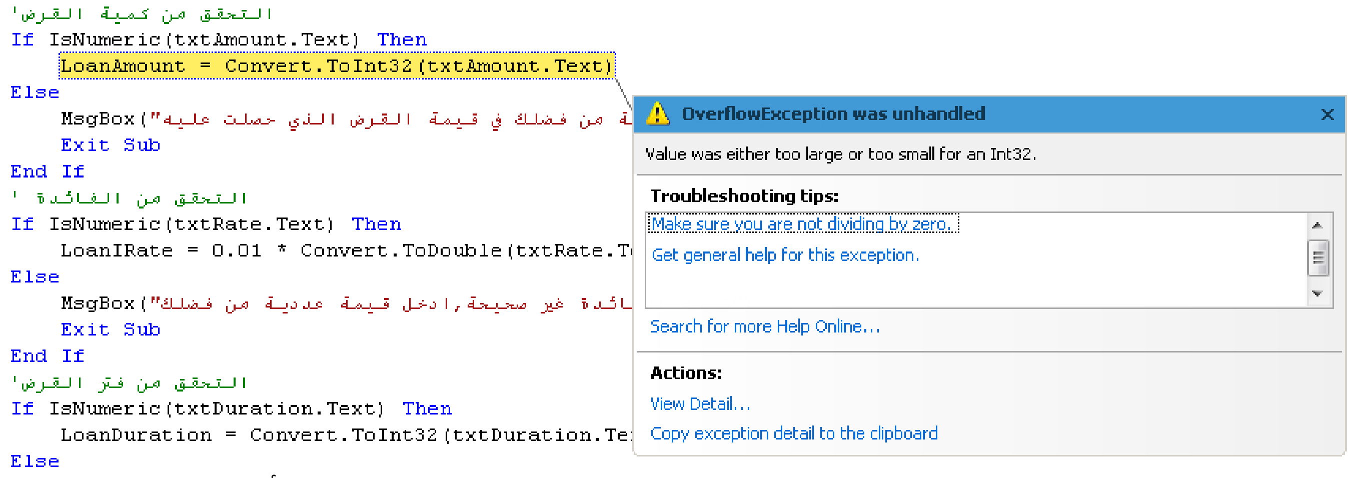
كما ذكرنا سابقا اذا ما عمل المستخدم على إدخال نص ما في صناديق النصوص التي على الفورم مثلا ادخل المستخدم القيمة النصية التالية "عشرون" فان ذلك سيؤدي الى حصول خطأ وسينهار البرنامج مباشرة عارضا رسالة خطأ, جرب هذا واقرأ رسالة الخطأ, لذا لمنع المستخدم من إدخال قيم غير مناسبة
الكود المعدل التالي يعمل على التحقق من بيانات المستخدم في كل صندوق نص لمنع إدخال بيانات غير مناسبة ☹(الكود يشرح نفسه وليس بحاجة الى أي شرح)

```
Dim Payment As Double
    Dim LoanIRate As Double
    Dim LoanDuration As Integer
    Dim LoanAmount As Integer
```

```
        ' التحقق من كمية القرض
        If IsNumeric(txtAmount.Text) Then
            LoanAmount = Convert.ToInt32(txtAmount.Text)
        Else
            MsgBox("ادخل قيمة عددية من فضلك في قيمة القرض الذي حصلت عليه")
            Exit Sub
        End If
        ' التحقق من الفائدة
        If IsNumeric(txtRate.Text) Then
            LoanIRate = 0.01 * Convert.ToDouble(txtRate.Text) / 12
        Else
            MsgBox("قيمة الفائدة غير صحيحة,ادخل قيمة عددية من فضلك")
            Exit Sub
        End If
        'التحقق من فتر القرض
        If IsNumeric(txtDuration.Text) Then
            LoanDuration = Convert.ToInt32(txtDuration.Text)
        Else
            MsgBox("من فضلك ادخل مدة القرض المناسبة, على ان تكون قيمة عددية")
            Exit Sub
        End If
        ' اذا كانت جميع البيانات صحيحة , باشر عملية الحساب
        Dim payEarly As DueDate
        If chkPayEarly.Checked Then
            payEarly = DueDate.BegOfPeriod
        Else
            payEarly = DueDate.EndOfPeriod
        End If
        Payment = Pmt(LoanIRate, LoanDuration, -LoanAmount, 0, payEarly)
        txtPayment.Text = Payment.ToString("#.00")
```

الوظيفة IsNumeric هي وظيفة أخرى جاهزة (معدة للاستخدام من قبل اللغة) وتقبل متغير وتعود بصح اذا كان المتغير عددي وتعود بخطأ في الحالات الأخرى.

والان جرب التطبيق حتى تفهم الكود ومن ثم ادخل قيمة هائلة في كمية القرض.وحاول ان تستكشف ما سيحدث ,سترى ان البرنامج ينهار ثانية ولكن مع رسالة خطأ مختلفة عن سابقتها,كما هو مبين في الشكل التالي تظهر رسالة خطأ معنونة بالعنوان التالي: "استثناء تجاوز الحد لم يتم معالجته" ويظهر مع الرسالة ايضا نصائح واقتراحات لمعالجة الخطأ . يشتكي الفيجوال بيسك من خطأ تجاوز الحد تحت عنوان رسالة الخطأ بالسطر التالي:" value was either too large or too small for an Int32 أي ان" القيمة أم ان تكون كبيرة جدا او صغيرة جدا من اجل المتغير من النوع الصحيح" وكما ترى فان البرنامج يتوقف عند السطر الذي سبب الخطأ ويعلمه باللون الاصفر,



تجاوز الحد overflowهي قيمة عددية كبيرة جدا لان يعمل البرنامج على معالجتها, وهذا الخطأ ينتج عادة عندما تقسم عدد ما على الصفر او على عدد صغير جدا, وعندما تسند قيمة كبيرة جدا الى متغير من نوع عدد صحيح ستحصل ايضا على خطأ تجاوز الحدoverflow ,في الحقيقة أي قيمة تتجاوز القيمة 2,147,483,647في تطبيق "حاسب القروض" ستسبب حالة تجاوز الحد,فهذه هي القيمة الأعلى التي تستطيع ان تسندها الى المتغير من النوع الصحيح, هذه القيمة مناسبة بشكل واسع لمتطلبات البنوك ولكنها غير كافية لمعالجة الإيرادات الحكومية والتي من المحتمل ان تتجاوز هذه القيمة.كما سترى في الفصل القادم يزودك الفيجوال بيسك بأنواع أخرى من المتغيرات التي تستطيع ان تخزن قيم هائلة (تجعل من national debt الدين الدولي (بين الدول) صغير عمليا) في الوقت الحالي اذا أردت ان تستخدم برنامج "حاسب القروض"صرح عن المتغيرLoanAmount كما يلي: Dim LoanAmount As Double حيث ان المتغير من نوع

المزدوج يستطيع ان يحفظ قيم كبيرة واكبر من القيم التي يحفظها المتغير من النوع الصحيح.وكان بالإمكان استخدام المتغير والتصريح عنه على انه من النوع Long الصحيح الطويل فهو ايضا يحفظ قيم اكبر منinteger الصحيح.
يمكن ايضا اصطياد خطأ تجاوز الحد من خلال كتابة كود التحقق من البياناتdata-validation code, فهناك على الدوام فرص لان ينتج عن حساباتك خطأ تجاوز الحد او أنواع أخرى من الأخطاء الرياضية ولكن التحقق من البيانات لن ينجح هنا,حيث انك لن تعلم النتيجة قبل ان تستكشف الحسابات, فنحتاج الى شيء ما يدعى معالج الخطأ error handling او مايسمى في الفيجوال بيسك" معالج الاستثناءexception handling " هذا الكود الإضافي يعالج الأخطاء عند وقوعها,في الحقيقة في هذه الحالة تخبر الفيجوال بيسك بان لا يتوقف عن العمل برسالة خطأ كالتي ظهرت معنا في المقاطع السابقة,والتي تربككembarrassing ولا تساعد المستخدم أبدا,فعوضا عن ذلك سيعمل الفيجوال بيسك على اصطياد الخطأ وينفذ العبارات المناسبة والتي تعالج الخطأ,.من الواضح ان عليك ان تكتب المزيد من الكود(عليك كتابة هذه العبارات التي تصطاد الخطأ).
يعمل مثال التطبيق هذا كمعلن عن الخطأ و بشكل اتوماتيكي عند حصول خطأ ما fail-safe في البرنامج,اذاً يتبقى علينا إضافة لمسة أخيرة على تطبيقنا,فالتنوع في القيم التي تكون على الفورم لا تكن دائما متوافقة (synch في وضع متزامن مع بعضها البعض) مثلا لنقول انك عملت على حساب " المستوجب دفعه شهريا" من اجل قرض معين,ومن ثم أردت ان تغير فترة القرض لترى كيف تؤثر على "المستوجب دفعه شهريا" فحالما تغير الفترة للقرض وقبل ان تضغط على زر حساب المدفوع شهريا ,فان القيمة في صندوق المدفوع شهريا لا تستجيب لمعاملات parameters القرض,بشكل مثالي يتوجب على صندوق نص المستوجب دفعه كل شهر ان يمحى حالما يعمل المستخدم على تحديث واحدة من بيانات او بارومترات القرض,لذا لعمل هذا عليك إدخال العبارات التي تنظف اداة صندوق نص المتوجب دفعه كل شهر أي الأداة *txtPayment, ولكن السؤال هنا ما هو معالج الحدث المناسب لهذه العبارة؟ تطلق اداة صندوق النص* TextBox الحدث TextChangedفي كل مرة يتم تغير نصها,وهذا هو المكان المناسب لتنفيذ العبارة التي تعمل على تنظيف اداة صندوق نص "المتوجب دفعه كل شهر " على الفورم,وبسبب وجود ثلاث أدوات صندوق نص على الفورم,عليك كتابة كود TextChanged "تغير النص لها جمعا" او كتابة معالج حدث يعالج الجميع دفعة واحدة كما يلي☹(في الكود التالي وضعت علامة أمام خاصية التفاف النص لذا تم تقطيع السطر الطويل من الكود الى سطرين كما تعلمت سابقا)

```
  Private Sub txtAmount_TextChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
txtAmount.TextChanged, txtDuration.TextChanged, txtRate.TextChanged
        txtPayment.Clear()
    End Sub
```

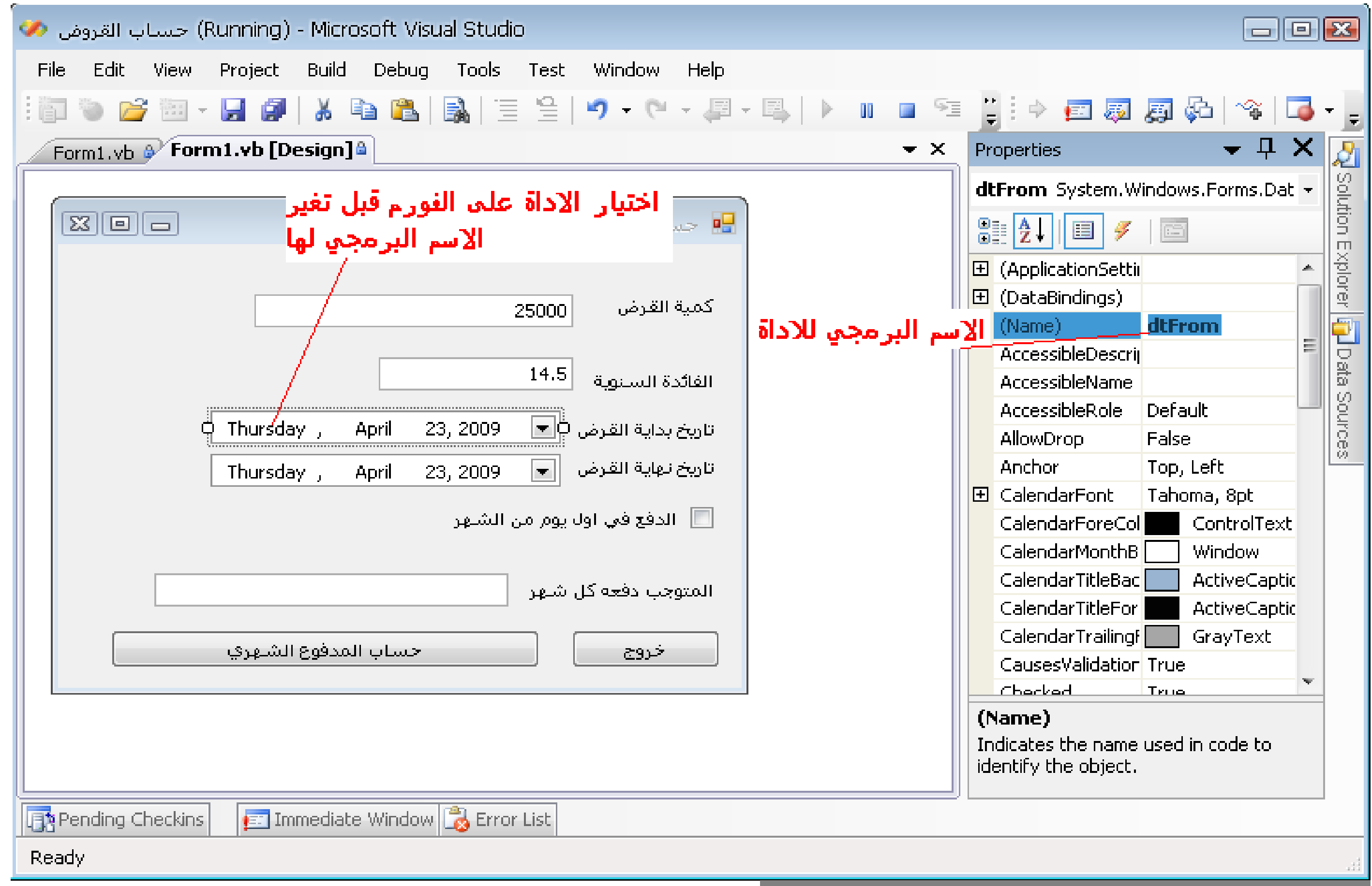
كما ترى فقد عملت على كتابة معالج عام لكل صناديق النصوص دفعة واحدة بدل من كتابة كلا على حدا وذلك بالضغط المزدوج على كل صندوق نص ومن ثم إدخال الكود txtPayment.Clear() في كل منها وبما ان الأحداث من نفس النوع فيمكن كتابتها جميعها بعد الكلمة المحجوزة Handles.
ومن امثلة المشاريع لهذا الفصل مشروع(او نسخة أخرى من مشروع حاسب القروض) تستخدم التاريخ بدل من فترة القرض بالأشهر وبالتالي يستخدم واجهة مختلفة,فبدلا من تحديد فترة القرض بالأشهر,يزود هذا التطبيق بنسختين من اداة فرز التاريخ والوقت DateTimePickerوالتي تستخدم لتحديد التاريخ, اعمل على حذف صندوق نص الفترة (شهر) من خلال النقر عليه ومن ثم الضغط على delete وكذلك احذف اداة العنوان الموفقة "الفترة (شهر)" وضع أداتي عنوان Labelsعلى الفورم و أداتي فرز التاريخ DateTimePicker فبوضع هذين الأداتين يستطيع المستخدم ان يدخل تاريخ بدء وانتهاء القرض ويقوم البرنامج بحساب فترة القرض بالشهور بالعبارات التالية:

```
LoanDuration = DateDiff(DateInterval.Month, dtFrom.Value, dtTo.Value) + 1
```

حيث ان dtFromهي خاصة الاسم البرمجي name لأداة فرز تاريخ بداية القرض وdtTo هي ايضا خاصية الاسم البرمجي name لأداة فرز تاريخ نهاية القرض أما الوظيفة DateDiff فتعطي الفرق بين التاريخين بالشهور وكما يحددها المعامل النسبي الأول للوظيفة ،أما المعاملات النسبيان الباقيان فيحدد الأول بداية الفترة ويحدد المعامل الثاني نهاية الفترة, مع الأخذ بعين الاعتبار الشهر الأول من القرض لذا تم إضافة 1 الى عملية حساب فترة القرض.
بعد ان تعمل التغيرات ستصبح واجهة التطبيق مشابهة للشكل التالي:

حساب المدفوع الشهري لقرض ما
كمية القرض 25000
الفائدة السنوية 14.5
تاريخ بداية القرض Wednesday, July 01, 2009
تاريخ نهاية القرض Tuesday , March 01, 2011
الدفع في اول يوم من الشهر
المتوجب دفعه كل شهر 1355.04
حساب المدفوع الشهري
خروج

ملاحظة :
خاصية الاسم البرمجي name هامة جدا فاسم الأداة الذي تكتبه في هذه الخاصية هو نفسه الذي يستخدم في الكود وتستطيع تغير الاسم البرمجي لأي اداة من نافذة الخصائص حيث تختار الأداة اولاً من على الفورم وكما هو موضح في الشكل التالي:



Building a Calculator بناء تطبيق الحاسبة

ستتعلم في هذا التطبيق كيفية بناء آلة حاسبة وسترى كيف يعمل الفيجوال بيسك على تبسيط العمليات المتقدمة , وعملية بناء التطبيق ليست بالتعقيد الذي قد يظنه الكثير من المبتدأين .

يضاهي هذا التطبيق الحاسبة الآلية اليدوية ويعالج العمليات الرياضية الأساسية,فله مظهر الحاسبة الرياضية,وتستطيع توسيع ميزات التطبيق بإضافة ميزات مثل الجيب و التجيب والظل و التظل وغيرها من العمليات الرياضية الأخرى

Designing the User Interface تصميم واجهة المستخدم

واجهة التطبيق بسيطة ,ولكنها مع ذلك تأخذ بعض الوقت والجهد,فعليك ترصف الأزرار على الفورم وتجعل من الحاسبة تشبه كثيرا الحاسبة اليدوية a hand-held calculator لذلك أبدا مشروع جديد وسمه " الحاسبة الرياضية" واعد تسمية الفورم الرئيسية له من Form1 الى **frmCalculator** من خلال خاصية الاسم البرمجي (name) ومن خلال خاصية النص text للفورم اكتب النص "الحاسبة"

لتبسيط عملية التصميم اتبع الخطوات التالية:

1- اختار الخط الذي تريده للفورم ،فجميع الادوات التي ستضعها على الفورم سترث هذا الخط, واجعل خاصية من اليمين الى اليسار صح وكذلك اجعل تخطيط النموذج من اليمين الى اليسار "نعم " وذلك من اجل ان تكون واجهة التطبيق تدعم اللغة العربية.

2- أضف اداة Labelعنوان, والتي ستصبح شاشة الحاسبة,وضع خاصية BorderStyle تخطيط الحدود الى Fixed3Dثلاثي الأبعاد غير خاصية لون الخط Fore Color وخاصية لون الخلفية BackColor كما تراه مناسب لتظهر الشاشة بشكل مختلف عن باقي الحاسبة.

3- اسحب اداة زر الى الفورم وغير خاصية Textالنص لها الى 1 و أعطه اسم برمجي name الى **btn1 انسخ هذا الزر والصقه تسع مرات على الفورم وضع خاصية النص لباقي الأزرار على التوالي الى 2و3و4و5و6و7و8و9و0 والاسم البرمجي** name **على التوالي الى btn2 وbtn3 وbtn4 وbtn5** و **btn6 وbtn7 وbtn8 وbtn9 وbtn0** قبل أن تقوم بعملية النسخ اعمل على تنسيق الزر الأول بأفضل تنسيق .

4- عملية النسخ واللصق كما قلنا سيكون لها نفس الاسم للزر المنسوخ لذا اعمل على تغير كل زر تقوم بلصقه الى العنوان المناسب وكما تم ذكره في الفقرة السابقة.

5- عندما تنتهي من أزرار الأرقام ضع زرين ايضا على الفورم احدهما الفاصلة و Nameاسمه البرمجي **btnperiod**أما خاصية text النص له فاكتب عليها النقطة فقط وهنا اجعل خاصية الخط كبيرة نوعا ما لهذا الزر كي تظهر النقطة بشكل واضح, والزر الثاني اكتب عليه من خلال خاصية text النص **c** فقط حيث سنستخدم هذا الزر لتنظيف الشاشة, واجعل Nameاسمه البرمجي **btnclear**.

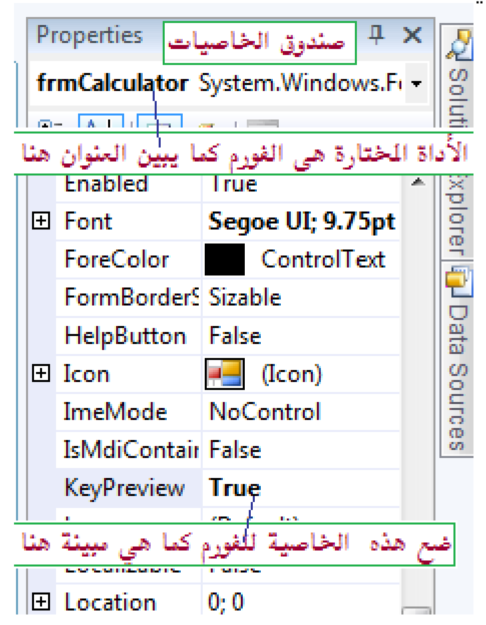
6- والآن ضع أزرار من اجل العمليات الرياضية على الفورم, الجمع (+) addition والطرح(−)subtraction والضرب (*) multiplication والقسمة(/) division والأسماء البرمجية لها هي على الترتيب(الجمع btnaddition,الطرح btnsubraction ,الضرب btnmultiplication,القسمةbtndivision).

7- أخيرا ضع زر المساواة على الفورم واجعله ومدده بقدر زرين واجعل خاصية النص له الى" = " وأسمه البرمجي إلى btnequal

8- نسق الأزرار على الفورم بحيث تبدو مشابهة للشكل التالي وضع كما هو مبين في الشكل زر لقلب الرقم واسمه البرمجيbtnreverse أما خاصية النص فضعها كما هي مبينة في الشكل إلى1/×,وأيضا أضف زر أخر لعكس الإشارة وضع خاصية النص إلى+- وخاصية الاسم البرمجي إلىbtnref كما هو مبين في الشكل :



ضع الخاصية التالية للفورم إلى صح لتتمكن من إظهار الأرقام عند برمجة الأحداث التابعة كبس الزر من خلال لوحة المفاتيح كما هو مبين في الشكل التالي



برمجة الحاسبة

قبل البدء بمناقشة برمجة الحاسبة سأقدم لك كود التطبيق كامل:

```
Public Class frmCalculator
    Dim clearDisplay As Boolean
    Dim Operand1 As Double
    Dim Operand2 As Double
    Dim MathOperator As String
    Private Sub DigitClick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btn0.Click, btn1.Click, _
        btn2.Click, btn3.Click, btn4.Click, btn5.Click, btn6.Click, btn7.Click, btn8.Click, btn9.Click
        If clearDisplay Then
            lblDisplay.Text = ""
            clearDisplay = False
        End If
        lblDisplay.Text = lblDisplay.Text + CType(sender, Button).Text
        btnequal.Focus()
    End Sub
    Private Sub bttnClear_Click(ByVal sender As System.Object, ByVal e  _
As System.EventArgs) Handles btnclear.Click
        lblDisplay.Text = ""
        btnequal.Focus()
    End Sub
    Private Sub bttnPeriod_Click(ByVal sender As System.Object, ByVal e _
 As System.EventArgs) Handles btnperiod.Click
```

```vb
        If lblDisplay.Text.IndexOf(".") > 0 Then
            btnequal.Focus()
            Exit Sub
        Else
            lblDisplay.Text = lblDisplay.Text & "."
            btnequal.Focus()
        End If
    End Sub

    Private Sub bttnPlus_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnaddition.Click
        Operand1 = Convert.ToDouble(lblDisplay.Text)
        MathOperator = "+"
        clearDisplay = True
        btnequal.Focus()
    End Sub
    Private Sub bttnEquals_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)_ Handles btnequal.Click
        Dim result As Double
        Operand2 = Convert.ToDouble(lblDisplay.Text)
        Try
            Select Case MathOperator
                Case "+"
                    result = Operand1 + Operand2
                Case "-"
                    result = Operand1 - Operand2
                    Debug.WriteLine("DD")
                Case "*"
                    result = Operand1 * Operand2
                Case "/"
                    If Operand2 <> "0" Then result = Operand1 / Operand2
            End Select
            lblDisplay.Text = result.ToString
            clearDisplay = True
        Catch exc As Exception
            Debug.WriteLine(exc.Message)
            result = "ERROR"
        End Try
    End Sub
    Private Sub bttnMinus_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)_ Handles btnsubraction.Click
        Operand1 = Convert.ToDouble(lblDisplay.Text)
        MathOperator = "-"
        clearDisplay = True
        btnequal.Focus()
    End Sub
    Private Sub bttnMultiply_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)_ Handles btnmultiplication.Click
        Operand1 = Convert.ToDouble(lblDisplay.Text)
        MathOperator = "*"
        clearDisplay = True
        btnequal.Focus()
    End Sub
    Private Sub bttnDivide_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)_ Handles btndivision.Click
        Operand1 = Convert.ToDouble(lblDisplay.Text)
        MathOperator = "/"
        clearDisplay = True
        btnequal.Focus()
    End Sub
    Private Sub bttnNegate_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)_ Handles btnref.Click
        lblDisplay.Text = -Convert.ToDouble(lblDisplay.Text).ToString
        clearDisplay = True
        btnequal.Focus()
    End Sub
    Private Sub bttnReverse_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)_ Handles btnReverse.Click
        If Convert.ToDouble(lblDisplay.Text) <> 0 Then
            lblDisplay.Text = (1 / Convert.ToDouble(lblDisplay.Text)).ToString
            clearDisplay = True
        End If
        btnequal.Focus()
    End Sub
    Private Sub CalculatorForm_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles
Me.KeyPress
        Select Case e.KeyChar
            Case "1" : btn1.PerformClick()
```

```
            Case "2" : btn2.PerformClick()
            Case "3" : btn3.PerformClick()
            Case "4" : btn4.PerformClick()
            Case "5" : btn5.PerformClick()
            Case "6" : btn6.PerformClick()
            Case "7" : btn7.PerformClick()
            Case "8" : btn8.PerformClick()
            Case "9" : btn9.PerformClick()
            Case "0" : btn0.PerformClick()
            Case "." : btnperiod.PerformClick()
            Case "C", "c" : btnclear.PerformClick()
            Case "+" : btnaddition.PerformClick()
            Case "-" : btnsubraction.PerformClick()
            Case "*" : btnmultiplication.PerformClick()
            Case "/" : btndivision.PerformClick()
            Case "=" : btnequal.PerformClick()
        End Select
    End Sub
    Private Sub CalculatorForm_Keydown(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyEventArgs) Handles
Me.KeyDown
        If e.KeyData = Keys.Enter Then
            btnequal.PerformClick()
            e.Handled = True
        End If
    End Sub
End Class
```

أنت الآن جاهز لإضافة بعض الكود الى التطبيق ,اضغط مزدوج على واحد من أزرار الأرقام وسترى الكود التالي في نافذة محرر الكود:

```
Private Sub btn1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btn1.Click, btn0.Click,
btn2.Click, btn3.Click, btn4.Click, btn5.Click, btn6.Click, btn7.Click, btn8.Click, btn9.Click

 End Sub
```

كما ترى فقد عملت على إضافة جميع أزرار الأرقام لأنها جميعا تشترك بهذا الحدث فلا يوجد داعي لكتابة جميع الأحداث لكل رقم على حدا لذا عملت على كتابتهم دفعة واحدة , فقط ضع فاصلة في نهاية السطر الأول من الكود أي بعد btn1.Click ومن ثم اختار اسم الزر ومن ثم نقطة واسم الحدث ضغط او Click.

أنت تعلم انك عندما تضغط زر رقم ما في الحاسبة اليدوية يتم إلحاق الرقم في الشاشة ,لمضاهاة هذه العملية ادخل السطر التالي في معالج Clickحدث الضغط على الأزرار ☹(أحيانا قد ترد Label1 وهي نفس lblDisplay ولكن أنت أعد تسمية أداة العنوان كما بينت سابقا)

```
Label1.Text = Label1.Text + sender.Text
```

هذا السطر يعمل على إلحاق الرقم الذي يتم ضغطه بشاشة الحاسبة,يمثل المعامل النسبي sender لحدث الضغط Clickالأداة التي تم ضغطها(او الأداة التي أطلقت الحدث) وخاصية النص Textلهذه الأداة هي نص الزر الذي تم ضغطه,فمثلا فإذا أدخلت بشكل مسبق القيمة 345 فان ضغط الرقم 0 سيعرض القيمة 3450 على اداة العنوان والتي تتصرف كشاشة عرض للحاسبة.

التعبير Sender.Text ليس هو الطريقة الأفضل   للتمكن من الوصول الى خاصية النص للزر الذي تم ضغطه, ولكنه سيعمل طالما ان خيار التدقيق غير فعالStrict option is off ,وكما تم مناقشته في الفصل الثاني علينا تحويل الكائن *sender*الى نوع معين(من نوع Buttonالزر) ومن ثم استدعاء طريقة Textالنص له لذلك يصبح الكود السابق كمايلي:

```
Label1.Text = Label1.Text + CType(sender, Button).Text
```

الكود خلف أزرار الأرقام يحتاج الى العديد من السطور ,فبعد عمل ما, يجب أن يتم تنظيف الشاشة,وذلك بعد الضغط على احد أزرار العمليات الرياضية مثل الجمع او الطرح او غيرها من العمليات,فالشاشة يجب أن يتم تنظيفها في انتظار in anticipationالطرف الثاني من العملية second operand, في الواقع يجب ان يتم تنظيف الشاشة حالما يتم الضغط على الرقم الأول التابع للطرف الثاني من العملية,وليس حالما يتم الضغط على زر العملية الرياضية,وبطريقة مماثلة يجب ايضا تنظيف الشاشة بعد ان يضغط المستخدم على زر المساواة , لذلك عدل Reviseالكود في معالج حدث النقر على الأزرار كما هو مبين(في الكود الكامل الذي قدمته لك سابقا كامل)

المتغير *clearDisplay* تم التصريح عنه كمتغير من النوع المنطقي Boolean وهذا يعني انه يأخذ واحدة من القيمتين أما صح او خطأ True or False,افرض ان المستخدم قد أنجز عملية ما والنتيجة ما تزال معروضة على شاشة الحاسبة,والمستخدم الآن يكتب عدد أخر,فبدون عبارة الشرط سيستمر البرنامج في إلحاق الأرقام الى الرقم الموجود مسبقا على الشاشة,.وهذه ليست الكيفية التي تعمل بها الحاسبة,عندما يبدأ المستخدم بإدخال عدد جديد فان الشاشة يجب ان تصبح فارغة, وبرنامجنا يستخدم المتغير *clearDisplay*لمعرفة متى علينا ان ننظف الشاشة.

يعمل زر المساواة على وضع المتغير clearDisplayالى صح ليدل على ان الشاشة تحوي على نتيجة عملية ما,فالإجراء الجزئي ()DigitClickضغط الزر يتفحص قيمة المتغير في كل مرة يتم فيها الضغط على زر جديد ,فإذا كانت القيمة صح فان حدث ()DigitClickضغط زر ما يعمل على تنظيف الشاشة,ومن ثم يطبع الرقم الجديد فيها,ومن ثم فان هذا الإجراء الجزئي يعمل على إسناد القيمة خطأ الى المتغيرclearDisplay لذلك عندما يتم ضغط الرقم التالي فان البرنامج لن يمسح الشاشة مرة أخرى,.

ولكن ماذا اذا ما عمل المستخدم خطأ ما ,و أراد التراجع عن الإدخال الأخير,؟ فالحاسبة اليدوية النموذجية hand-held calculator ليس لديها زر Backspace "التراجع ضغطة للخلف" وزرClear التنظيف يمحي العدد الحالي الذي على الشاشة,لنعالج هذه الميزة,اضغط مرتين على الزر C وادخل الكود في معالج حدث الضغط عليه(بينته في الكود الكامل)
نستطيع الآن النظر إلى زر الفاصلة(النقطة),يعمل زر النقطة تماما مثل أزرار الأرقام,مع استثناء وحيد هو أنه يمكن للرقم أن يظهر أي عدد من المرات في قيمة رقمية,ولكن النقطة يمكن أن تظهر مرة واحدة فقط في قيمة عددية. فعدد مثل 99.991 هو ممكن ولكن عليك التأكد من أن المستخدم ليس بإمكانه إدخال عدد مثل التالي**23.456.55** بعد إدخال النقطة يجب على هذا الزر أن لا يعمل على إدخال نقطة أخرى(فاصلة أخرى فلن تحتاج لرؤية رقم فيه أكثر من فاصلة واحدة وغير ذلك فالرقم غير صحيح)(لاحظ الكود الذي يعالج هذه الحالة في حدث الضغط على زر الفاصلة)
IndexOfهي طريقة يمكن تطبيقها على أي سلسلة نصية.التعبير lblDisplay.Text هو نص (النص الذي على أداة العنوان) لذا نستطيع استدعاء طريقتها IndexOf, يعود التعبير (".")lblDisplay.Text.IndexOf بمكان النسخة الأولى من الفاصلة في نص أداة العنوان, فإذا كان هذا العدد(الذي ترجعه الطريقة (".")lblDisplay.Text.IndexOf) هو صفر أو عدد موجب, فالعدد الذي تم إدخاله يحتوي مسبقا على فاصلة,وفاصلة أخرى لايمكن إدخالها, في هذه الحالة يخرج البرنامج من الإجراء.أما إذا أعادت الطريقة السابقة 1- فإنه يتم إلحاق الفاصلة إلى العدد حتى الآن تماما كالأرقام النظامية, لقد عملنا فيما مضى واجهة المستخدم التي تضاهي الحاسبة اليدوية مع إمكانية إدخال البيانات,ولكنها لا تقوم بأي عملية حتى الآن .

برمجة العمليات الرياضية Coding the Math Operations

نستطيع الآن الانتقال إلى إدخال الجزء الهام من التطبيق:برمجة العمليات الرياضية ,لنبدأ بتعريف ثلاث متغيرات:

```
Dim Operand1 As Double لحفظ الرقم الأول في العملية
Dim Operand2 As Double لحفظ الرقم الثاني في العملية
Dim MathOperator As String...لاختيار العملية التي ترغب بإجراءها هل هي جمع ,قسمة
```

عندما يضغط المستخدم واحد من الرموز الرياضية,فإن القيمة التي على الشاشة يتم تخزينها في المتغير الأولOperand1 فإذا ضغط المستخدم زر الجمع يجب أن يعمل ملاحظة لنفسه أن العملية الحالية هي الإضافة ويعمل على وضع المتغير *clearDisplay*إلى صح,بحيث يستطيع المستخدم أن يدخل قيمة أخرى (القيمة الثانية التي يجب إضافتهاOperand2).تم تخزين رمز العملية في المتغير MathOperator,.فالمستخدم سيدخل قيمة ثانية ومن ثم يضغط على زر المساواة ليرى النتيجة.عند هذه النقطة يجب أن يعمل برنامجنا التالي:

1- قراءة القيمة التي على الشاشة في المتغير
2- إتمام العملية المشار إليها بواسطة المتغير MathOperator على متغيري الطرفيين Operand1 وOperand2
3- عرض النتيجة ووضع المتغير *clearDisplay*إلى صح True.

يجب أن يتم زر المساواة العملية التالية: Operand1 Operator Operand2
على فرض أن الرقم الذي على الشاشة عندما ضغط المستخدم على زر الجمع addition هو3342.ومن ثم أدخل المستخدم القيمة23 وضغط زر المساواة.يجب أن ينفذ البرنامج الجمع:23 + 3342 وإذا ضغط المستخدم زر القسمة فالعملية هي كما يلي: 23 / 3342 المتغيرات هي متغيرات محلية في الإجراء الذي تم التصريح عنهم فيه,والإجراءات الأخرى ليس لديها إمكانية الوصول إليها ولا تستطيع قراءة قيمها,في بعض الأحيان يجب أن تكون للمتغيرات إمكانية الوصول من العديد من الأماكن في البرنامج.فالمتغيرات MathOperator, *Operand2*, *Operand1*بالإضافة إلى المتغير *clearDisplay*يجب أن تكون من الممكن الوصول لها من ضمن أكثر من إجراء لذا يجب أن يتم التصريح عنها خارج يطاق أي إجراء ,فالإعلان عنهم عادة يتم في بداية الكود بالعبارات المبينة في الكود السابق وكما هو موضح في (بداية مقطع برمجة العمليات الرياضية)

```
Dim clearDisplay As Boolean
Dim Operand1 As Double
Dim Operand2 As Double
Dim MathOperator As String
```

هذه المتغيرات تم استدعاءها من متغيرات على نطاق واسع,أو ببساطة من أي إجراء على الفورم,لنرى كيف يستخدم البرنامج المتغير MathOperatorعندما يضغط المستخدم على زر الجمع,يجب أن يخزن البرنامج القيمة"+"في المتغيرMathOperator .وهذا يحدث من ضمن حدث نقر زر الجمع.جميع المتغيرات التي تخزن قيم عددية تم التصريح عنها كمتغيرات من النوع المزدوج Double,والتي بإمكانها تخزين قيم بأكبر دقة ممكنة.والمتغير المنطقي Booleanيأخذ قيمتين:صح Trueوخطأ False وقد رأيت مسبقا كيف يتم استخدام المتغير clearDisplay.
تم إسناد المتغير Operand1إلى القيمة الحالية التي على الشاشة.الطريقة ()Convert.ToDoubleتحول معاملها النسبي إلى قيمة مزدوجة. Text خاصية النص لأداة العنوان Labelمن نوع السلسلة الحرفيةstring. القيمة الفعلية التي تم تخزينها في خاصية النص هي في الحقيقة ليست عدد..إنها سلسلة حرفية stringمثل 428 والتي تختلف عن القيمة العددية 428 .وهذا هو السبب الذي دعانا إلى استخدام الطريقة Convert.ToDouble لتحويل قيمة العنوان لأداة العنوان إلى قيمة عددية.أما بالنسبة لأزرار العمليات الأخرى فإنها تفعل المثل وهي موضحة في الكود. .
بعد إدخال الطرف الثاني ,يستطيع المستخدم أن يضغط على زر المساواة لحساب النتيجة,عندما يحدث هذا فان الكود خلف زر المساواة يتم تنفيذه(انظر الكود خلف هذا الزر).
تم التصريح عن التغير *result*من النوع المزدوج Doubleلحفظ نتيجة عملية الحساب وبالدقة الأعلى.يستخرج الكود القيمة المعروضة في أداة العنوان ويخزنها في المتغير *Operand2*.ومن ثم فإن الكود يعمل على إتمام العملية بالعبارة "اختر حالةSelect Case "وهذه العبارة تقارن قيمة المتغير بالنسبة للقيم المجدولة بعد كل عبارة حالة.فإذا ما طابقت قيمة المتغير MathOperator قيمة إحدى الحالاتCase, فإنه يتم تنفيذ الحالة الموافقة(جمع أو طرح أو......)كما هو مبين في الكود خلف زر المساواة.
تأخذ عملية القسمة بعين الاعتبار قيمة متغير الطرف الثاني للعملية لأنه إذا كان صفر فلا يمكن تنفيذ العملية .العملية الأخيرة تنفذ القسمة فقط إذا كان المقام ليس صفر.فإذا حدث لأن يكون المتغير *Operand2* صفر فلا يحدث شيء.
والآن شغل التطبيق وتفحصه,ترى أنه يعمل مثل الحاسبة اليدوية,ولا تستطيع جعله ينهار بكتابة بيانات غير صحيحة.فليس علينا استخدام أي كود للتحقق من البيانات في هذا المثال لأن المستخدم ليس لديه فرصة لكتابة بيانات غير صحيحة.فآلية إدخال البيانات سهلة جدا foolproof أو مضمون.لا يستطيع المستخدم إلا إدخال بيانات عددية لأنه لا يوجد إلا أرقام عددية على الحاسبة.والخطأ الوحيد المحتمل هو القسمة على صفر ,وقد تمت معالجته في زر المساواة.وبالطبع سيكون بإمكان المستخدم كتابة فقط القيم العددية ولن تجبرهم على ضغط الأزرار التي على الحاسبة لكتابة أرقامهم فيجب أن تمكن المستخدمين من كتابة الأرقام التي تكون على لوحة المفاتيح,لحصر (أو اعتراض) كبس الأزرار keystrokesضمن كودك عليك أولا وكما بينت سابقا وضع الخاصية KeyPreview للفورم إلى صح. يتم تبليغ كل عملية كبس زر keystrokesإلى الأداة التي عليها التركيز في الوقت المعني وتطلق كبس الزر المعتد على الأحداث: الأحداث (المفتاح للأعلىKeyDown, ضغط الزر KeyPress,المفتاح للأسفل KeyUp).نحتاج في بعض الأحيان إلى معالجة كبس أزرار محددة من مكان رئيسي(مركزي),لذلك نضع خاصية الفورم KeyPreviewإلى صح,لذا فإن كبس الأزرار يتم تبليغه أولاً إلى الفورم ومن ثم إلى الأداة التي لديها التركيز.فنستطيع حصر (اعتراض)كبس الأزرار في حدث كبس الأزرار التابع

للفورم ومعالجتها في معالج الحدث هذا(راجع الكود الكامل وخاصة الإجراء الذي بدايته:Private Sub CalculatorForm_KeyPress) والذي يبين معالج حدث كبس الأزرار للفورم

معالج الحدث هذا يتفحص المفتاح الذي تم كبسه من قبل المستخدم ويناشد معالج حدث نقر Clickالزر للزر buttonالمناسب باستدعاء طريقته PerformClick.هذه الطريقة تتيح لك "نقر"click "زر من ضمن كودك. عندما يضغط المستخدم على الرقم 3 فإن معالج الحدث "كبس زر KeyPress " التابع للفورم يحصر كبس المفاتيح keystrokesويعمل على مضاهاتها مع الزر btn3.

استخدام أدوات التصحيح البسيطة Using Simple Debugging Tools

تعمل تطبيقاتنا التي قمنا بها حتى الآن بشكل جيد وهي سهلة التجريب والإصلاح إذا وجدت خطأ ما(لأنها تطبيقات بسيطة جداً)عندما تكتب كود ما ستكتشف حالاً أن شيء ما لا يعمل كما هو متوقع منه تماما,وعليك أن تكون قادر على اكتشاف لماذا,ومن ثم اصلاح الخلل.عملية التخلص من الأخطاء تدعى إزالة الأخطاء *debugging* وتوفر الفيجوال استوديو أدوات لتبسيط معالجة الأخطاء,يوجد العديد من تقنيات التصحيح debugging البسيطة التي يجب أن تعرفها حتى ولو كنت تعمل مع تطبيقات بسيطة.

افتح تطبيق الحاسبة واذهب إلى محرر الكود وضع المؤشر في السطر الذي يحسب الفرق بين طرفين.لنتظاهر(ندعي)انه يوجد مشكلة ما في هذا السطر ونريد تتبع تنفيذ البرنامج بشكل أقرب لاكتشاف الخطأ الذي فيه.اضغط F9وسيبرز highlightedالسطر باللون البني.هذا السطر قد أصبح breakpoint نقطة توقف:فحالما يصل التنفيذ إليها سيتوقف البرنامج عندها.

اضغط F5لتشغيل التطبيق وأكمل عملية طرح.أدخل عدد ومن ثم انقر على زر الطرح ومن ثم عدد أخر ,وأخيراً زر مساواة.سيتوقف التطبيق ومحرر الكود سيفتح .سيتم إبراز نقطة التوقف باللون الأصفر. وأنت ما تزال في صيغة (نظام)وقت التشغيلruntime mode ,ولكن تنفيذ التطبيق تم إيقافه مؤقتا(تأجيله بشكل مؤقتsuspended)تستطيع حتى تحرير editالكود في نظام الإيقاف break mode المؤقت وضغط المفتاح F5لمتابعة تنفيذ التطبيق.مهما يكن التأشير فوق المتغير *Operand1*والمتغير *Operand2*في نافذة محرر الكود,فان القيمة للمتغير الموافق ستظهر في صندوق المساعدةToolTip box . حرك المؤشر فوق أي متغير في معالج الحدث الحالي لترى قيم هذه المتغيرات.هذه هي قيم المتغيرات تماما قبل prior تنفيذ عبارة الإظهار(الإبرازhighlighted)

ضع المؤشر فوق المتغير *result* ترى أنه صفر لأن هذه العبارة لم يتم تنفيذها حتى الآن .إذا كانت المتغيرات involvedالمحتواة ضمن هذه العبارة لديها القيم المناسبة (إذا كانت هذه المتغيرات ليس لديها القيم المناسبة,عليك أن تعلم أن المشكلة priorسابقة لهذه العبارة وربما في معالج حدث آخر)تستطيع تنفيذ هذه العبارة بالضغط على F10,والذي ينفذ فقط العبارة المعلمة(المبرزةhighlighted)والبرنامج سيتوقف عند السطر التالي.العبارة التالية التي يجب أن يتم تنفيذها هي العبارة"أنهي الاختيارEnd Select "

أوجد حالة instanceمن المتغير *result*في معالج الحدث الحالي,وأعد restوضع المؤشر فوقها وسترى قيمة المتغير بعد أن تم اسناد قيمة له.تستطيع الآن الضغط على F10لتنفيذ عبارة أخرى أو الضغط على F5للعودة إلى نمط التنفيذ العادي.

تستطيع أيضا تقييم التعابير التي تحتوي involvingعلى أي متغيرات في معالج الحدث الحالي بإرسال العبارة المناسبة إلى نافذة الإظهار المباشرImmediate window (كما تعلمت مسبقا).تظهر نافذة الإظهار المباشر أسفل بيئة التطوير المتكاملة.فإذا كانت غير مرئية,افتح قائمة تصحيح Debugواختار نوافذ Windowsومنها نافذة الإظهار المباشرImmediate window . يتم سبق السطر الحالي في نافذة الأمر command window بالرمز "أكبر من >"(تذكار reminiscentمن أيام الدوس) ضع الموشيرة cursorبجانبه وأدخل العبارة التالية:

```
? Operand1 / Operand2
```

تستطيع إظهار نافذة الأمر من قائمة عرض view نوافذ أخرى other windowsومنه نافذة الأمرcommand window

quotientقسمة قيمتين ستظهر في السطر التالي.علامة القسمة هي فقط اختصار لرمز shorthand notationكتابة أمر.إذا كنت تريد أن تعرف القيمة الحالية على شاشة الحاسبة,أدخل العبارة التالية:

```
? lblDisplay.Text
```

هذه العبارة تطلب قيمة خاصية النص للأداة التي على الفورم.القيمة الحالية لنص أداة العنوان سيظهر في السطر التالي.تستطيع تقيم تعبير رياضي بعبارة مثل التالية:

```
? Math.Log(3/4)
```

()Logاللوغ هو وظيفة اللوغاريتم logarithmوهي طريقة من طرق فئة الرياضيات.Math class .حاليا,ستكتشف أن نافذة الإظهار المباشر أداة ملائمة للاستعمال handy tool من أجل تصحيح التطبيقات.فإذا كان لديك عبارة ذات تعبير معقد ,تستطيع أن تطلب قيم المكونات المستقلة للتعبير وتتأكد من صحة هذه المكونات.

والآن حرك المؤشر فوق نقطة التوقف واضغط مرة ثانية F9.هذا سيزيل علامة التوقف وتنفيذ البرنامج لم يتم قطعه في المرة القادمة عندما يتم تنفيذ هذه العبارة.

إذا لم يتم توقيف تنفيذ البرنامج عند نقطة توقف breakpoint هذا يعني أن نقطة التوقف لن يتم الوصول لها ابدا (كان تضع نقطة التوقف مثلا في حالة تابعة للجمع وتقوم بتنفيذ عملية الطرح فمن البديهي أن البرنامج سيتجاوز عملية الجمع ولن يختار هذه الحالة وبالتالي فإن نقطة التوقف لن يتم الوصول إليها أبداً).إذا لم تسند قيمة مناسبة للمتغيرMathOperator فإن جملة حالة عملية الطرح كما قلنا لن يتم الوصول لها أبدا. عليك وضع نقطة التوقف عند بداية عبارة قابلة للتنفيذ لمعالج حدث نقر زر المساواة لتجرب القيم لجميع المتغيرات لحظة بدء تنفيذ هذا الإجراء.إذا كان للمتغيرات القيم المتوقعة ,ستستمر بالتقدم في اختبار الكود .وإذا كانت القيم غير ذلك للمتغيرات(غير متوقعة)عليك اختبار العبارات التي تؤدي إلى هذه العبارة(العبارات في معالجات الأحداث للأزرار المتنوعة).

هناك تقنية أخرى بسيطة لمعالج أخطاء التطبيقات وهي طباعة قيم متغير معين في نافذة الإظهار المباشرImmediate window .على الرغم من أن هذه ليست أداة تصحيح.فهي معروفة لمبرمجي الفيجوال بيسك(وهي عملية جدا).العديد من المبرمجين يقومون بطباعة قيم متغيرات مختارة قبل وبعد تنفيذ بعض العبارات المعقدة.لفعل هذا استخدم العبارة: Debug.WriteLineمتبوعا باسم المتغير الذي تريد أن تطبع قيمته أو متبوعا باسم التعبير الذي تريد أن تظهر قيمته في نافذة لإظهار المباشر:

```
Debug . WriteLine (Operand1)
```

ترسل هذه العبارة مخرجاتها إلى نافذة الإظهار المباشر .وهذه تقنية بسيطة ,ولكنها تعمل .تستطيع أيضا استخدامها لاختبار استدعاء وظيفة أو طريقة .إذا كنت غير واثق من الشكل العام للوظيفة ,مرر تعبير مناسب يحتوي على وظيفة معينة إلى العبارة the Debug.WriteLine كمعامل نسبي.إذا ظهرت النتيجة المتوقعة في نافذة الإخراج المباشر,تستطيع أن تتقدم للأمام في استخدامها في كودك.

معالجة الأخطاء (الاستثناءات)Exception Handling

انهيار تطبيق الحاسبة لن يكون بسهولة انهيار تطبيق حاسب القروض.اذا بدأت بضرب عددين كبيرين جدا,فلن تحصل على خطأ تجاوز الحد overflow.ادخل عدد كبير بتكرار كتابة الرقم 9 ومن ثم اضربه برقم أخر قريب منه في الضخامة.عندما تظهر النتيجة انقر رمز الضرب مرة أخرى وأدخل رقم كبير جدا وتابع ضرب النتيجة بقيمة كبيرة جدا حتى تستهلك مجال القيمة المتاحة بالنسبة للمتغير من النوع المزدوج(حتى تصبح النتيجة كبيرة جدا بحيث لا يمكن تخزينها في متغير من النوع المزدوج)عندما يحدث هذا فان النص لا نهاية *infinity*سيظهر في شاشة الحاسبة.هذه هي طريقة الفيجوال بيسك التي تخبرك أنه ليس بإمكانها معالجة الأعداد الكبيرة.وهذا غير مقتصر على الفيجوال بيسك إنه أسلوب الكمبيوتر في تخزين القيم:حيث توفر عدد محدود من البايتات من أجل كل متغير.(لقد ناقشنا odditiesالشواذ مثل اللانهاية infinity في الفصل الثاني).لا نستطيع إنشاء خطأ تجاوز الحد بتقسيم عدد ما على الصفر,لأن الكود لن يقوم حتى بالمحاولة في تنفيذ هذا الحساب.باختصار.تطبيق الحاسبة إلى حد ما متين(نشيط robust) ,ولكن ومهما يكن لا نستطيع أن نكون واثقين من أن المستخدم لن يقوم بالتسبب في انتاج خطأ ما في التطبيق.لذا فعلينا توفير بعض الكود لمعالجة جميع أنواع الأخطاء .

ملاحظة الاستثناءات مقابل الأخطاء *Exceptions* versus *Errors*

تسمى الأخطاء *Errors*الآن الاستثناءات*exceptions* .بإمكانك أن تفكر بالأخطاء كاستثناءات في سياق التنفيذ الطبيعي .فإذا ما حدث استثناء,فيجب على البرنامج أن ينفذ عبارات خاصة لمعالجة الاستثناء وهذه العبارات لن يتم تنفيذها بالحالة الطبيعية.أظن أنه تم تسميتها استثناءات لأن الخطأ *error*كلمة لا يستسيغها أي انسان.ومعظم الناس لا يقبلون بأن بكتابة كود يحتوي على أخطاءerrors .فالمصطلح استثناء *exception*يمكن أن

يكون غامض(مبهمvague).فبدلا من أن تخبر الزبائن :أن التطبيق الذي كتبته فيه أخطاء ,تقول أن كودك قد أطلق استثناءraise an exception .ربما لم تتم ملاحظته,ولكن المصطلح *bug*خطأ لم يعد مستخدم بشكل متكرر بعد الآن .والأخطاء تدعى الآن قضايا معرفية *known issues* .ولكن مهما يكن مايزال المصطلح "إزالة الأخطاء *debugging* "كما هو ولم يتم تغيره.

كيف تعمل لتمنع ظهور استثناء ما حسابيا؟ فالتحقق من البيانات Data validationلن يساعد.حيث أنه من غير الممكن التنبؤ بنتيجة عملية ما تماما بدون إنجاز هذه العملية بشكل فعلي.وإذا سببت العلمية تجاوز الحدoverflow ,فلا تستطيع منعه.الجواب هو بإضافة معالج الاستثناء الهيكلي *structured exception handler*,معظم كود التطبيق بسيط جدا ومن الصعوبة توليد استثناء لتوضيح الأهداف.المكان الوحيد الذي يمكن أن يحدث فيه استثناء هو معالج زر المساواة .حيث تحدث الحسابات.وفي هذا المكان علينا إضافة معالج الاستثناء.الخطوط الرئيسية outlineلتركيب معالج الاستثناء هي التالية:

```
        Try
{مقطع العبارات }
          Catch Exception
{مقطع المعالجة }
          Finally
{تنظيف(إزالة أخطاء) مقطع العبارات}
      End Try
```

سيحاول البرنامج اتمام الحسابات التي تمت كتابة كودها في "مقطع العبارات" فإذا نجح البرنامج ,فإنه يتابع التنفيذ عند مقطع "تنظيف مقطع العبارات(إزالة الأخطاء) cleanup"وهذه العبارات على الأغلب عبارات إزالة الأخطاء,ومقطع "أخيراFinally " هو اختياري.فإذا تم فقده فلا توجد مشكلة وتنفيذ البرنامج سيستمر بالعبارات التي تلي العبارة"أنهي حاولEnd Try ".إذا حدث خطأ ما في المقطع الأول فإن مقطع "التقط الاستثناءCatch Exception " يتم تفعيله والعبارات في مقطع المعالجة هذا سيتم تنفيذه.

مقطع "التقط Catch " هو المكان الذي تعالج فيه الخطأ,ولا يوجد الكثير لتفعله فيما يتعلق بالأخطاء التي تنتج عن الحسابات.كل ما تستطيع فعله هو عرض تحذير وتمنح المستخدم فرصة لتغير القيم. مهما يكن يوجد نوع أخر من الأخطاء, تلك التي يمكن معالجتها بلباقة كبيرة جدا,فإذا كان برنامجك لا يستطيع القراءة من مشغل الأقراص المدمجة.تستطيع منح المستخدم فرصة لإدخال القرص المدمج والمحاولة مرةً أخرى.في الحالات الأخرى,تستطيع أن تطلب من المستخدم إدخال قيمة مفقودة ومن ثم الاستمرار.إذا حاول التطبيق مثلاً الكتابة على ملف "للقراءة فقطread-only " مثل تحديد ملف على القرص المدمج(وهذه الملفات تكون عادة للقراءة فقط ولا يمكن الكتابة عليها)أو ملف تكون مواصفة للقراءة فقط فعالة.تستطيع عرض تحذير ,والخروج من الإجراء الذي يحفظ البيانات ,وتمنح المستخدم فرصة إما لاختيار اسم ملف آخر أو تغير المواصفة للقراءة فقط بالنسبة للملف الذي تم اختياره.

بشكل عام ,لا توجد طريقة مفردة ومميزة لمعالجة كل الاستثناءات,يجب عليك أن تأخذ بعين الاعتبار جميع أنواع الاستثناءات التي يمكن أن يسببها تطبيقك ومعالجتها بقوام مستقلindividual basis .الأكثر أهمية بشأن معالجات الأخطاء هو أن لا ينهار تطبيقك,ببساطة لا يقوم بعمل العمليات التي تسبب الأخطاء (وهذا يعرف أيضا بعملية الإغاظة(التكدير*offending operation*) أو عبارات التجريح *offending statement*) ومن ثم يتابع التنفيذ بدون اية معالجة.معالج الخطأ من أجل تطبيق الحاسبة يجب أن يخبر (يبلغinform) المستخدم أن خطأ ما قد حدث ويخرج من الحسابات,ولا حتى يحاول عرض النتيجة,(راجع كود الحاسبة ولا حظ معالج الحدث الذي تم أضافته إلى معالج حدث النقر على زر المساواة)

يبقى معالج الخطأ غير فعال في أغلب الأوقات ولا يتدخل بعمليات البرنامج.فإذا ما حدث خطأ ما وعلى الأغلب سيكون خطأ تجاوز الحد overflow,فإن مقطع معالج الخطأ بالعبارات "حاول...التقط...انهي حاولTry. . .Catch. . .End Try" سيتم تنفيذه.يعرض هذا الكود صندوق رسالة تحتوي على وصف للخطأ وتعرض ايضا النص "خطأ ERROR" على شاشة الحاسبة. مقطع"أخيراً Finally" يتم تنفيذه بغض النظر عن حدوث الخطأ في هذا المثال يعمل مقطع"أخيراً Finally" على وضع قيمة المتغير clearDisplayإلى صح Trueلذلك عندما يتم ضغط زر رقم أخر فإن الرقم الجديد سيظهر على الشاشة.

الفصل الخامس
سيضاف فيمابعد إن شاء الله.(لا تقلق فهو لن يؤثر على السياق العام لأنه فصل خاص)

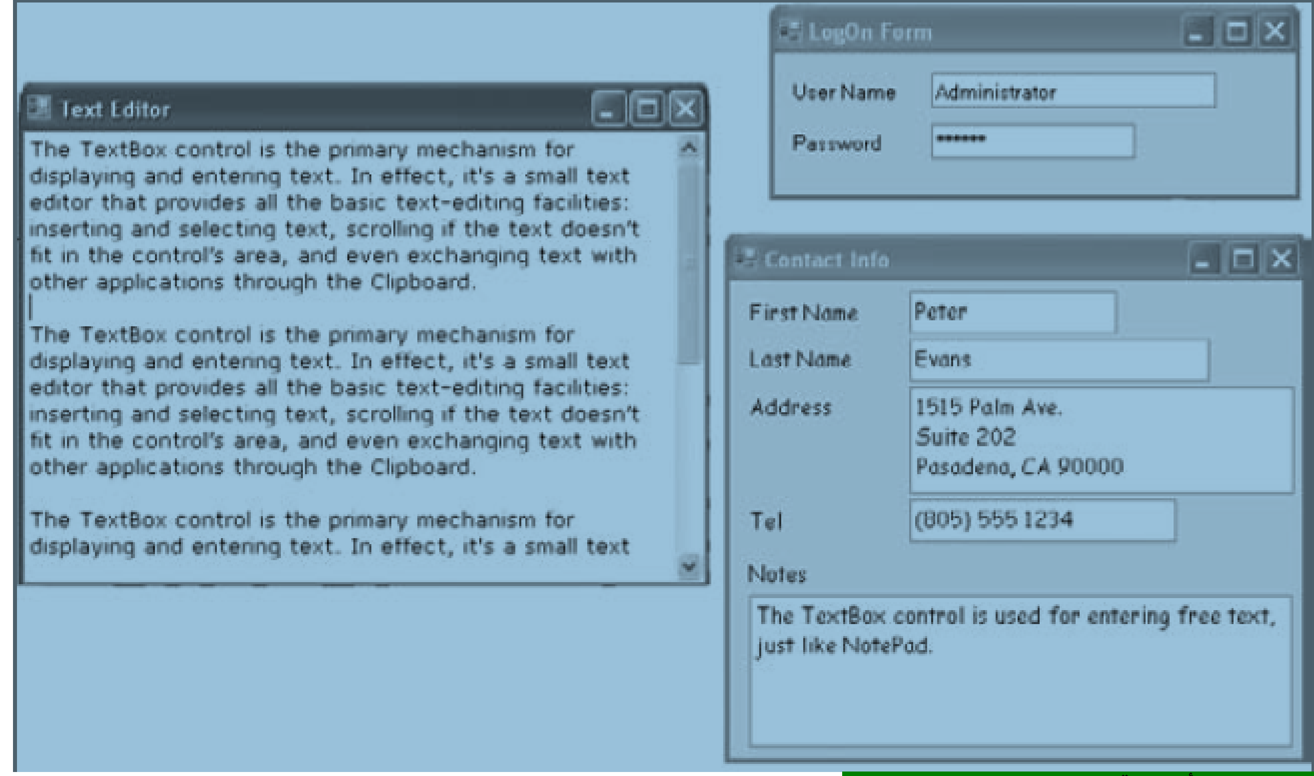
أدوات ويندوز الأساسية Basic Windows Controls

في الفصول السابقة قمنا باستكشاف بيئة تطوير الفيجوال بيسك ومبادئ البرمجة المقادة بالأحداث والتي هي جوهر نظام البرمجة للفيجوال بيسك. أثناء عمليات المعالجة عملنا على استكشاف بعض الأدوات الأساسية باختصار من خلال الأمثلة. يزودك إطار العمل بالعديد من الأدوات الأخرى ولكل منها multitude حشد من الخصائص العادية (مثل الخط,ولون الخلفية، وما الى ذلك)والتي تستطيع أن تعمل على إعدادها أما من خلال نافذة الخصائص أو من ضمن كودك.سنستكشف في هذا الفصل بعمق أدوات ويندوز الأساسية:الأدوات التي تستخدمها غالبا في تطبيقاتك لأنها بلوكات البناء الأساسية لواجهة المستخدم.ستتعلم في هذا الفصل مايلي:

-1 استخدام اداة TextBoxصندوق النص كأداة ادخل البيانات وأداة تحرير النص.
-2 استخدام اداة ListBox صندوق القائمة,وأداة CheckedListBox صندوق اختيار القائمة وأداة ComboBoxالصندوق المركب لتمثيل قوائم من البنود.
-3 استخدام اداة ScrollBarشريط الانزلاق,وأداة TrackBarشريط المسار(التزادف) لتمكن المستخدم من تحديد الحجم والموضع باستخدام الفأرة.

أداة صندوق النص The TextBox Control

إن اداة صندوق النص هي الآلية الرئيسية لعرض وإدخال النصوص.حيث انها محرر نص صغير يزود جميع التجهيزات الأساسية اللازمة لتحرير النصوص:مثل إدخال واختيار النص,الانزلاق اذا كان النص غير مرئي او لا يناسب المساحة,وحتى تبديل النص من تطبيقات أخرى من خلال الحافظة Clipboard. اداة صندوق النص هي اداة متعددة الاستعمالات extremely versatile وبشكل واسع فيما يخص إدخال البيانات حيث انك تستطيع ان تستخدمها لإدخال وتحرير السطور المفرد من نص ، مثل الأعداد او كلمات المرور,او ملف نصي كامل. وتستطيع ايضا اذا كانت مساحة النص غير كافية ان تعمل على إعداد خاصية شريط الانزلاق العمودي او الأفقي ويمكن ان تظهر هذه الأشرطة بشكل أوتوماتيكي عندما تصبح مساحة صندوق النص غير كافية لعرض النص في خاصية النص.جميع صناديق النصوص في الشكل التالي تحوي على نصوص بعضها ذات سطر مفرد والبعض الأخر متعدد الأسطر ومنها له شريط انزلاق كما هو مبين في الشكل التالي:



الخصائص الأساسية Basic Properties

لنبدأ مع الخصائص الأساسية التي تحدد appearanceالمظهر وبعض درجات التخصص الوظيفي لأداة صندوق النص، وهذه الخصائص عادة تعمل على إعدادها وقت التصميم من خلال نافذة الخصائصProperties window . ومن ثم فإننا سنلقي نظرة على الخاصيات التي تتيح لك معالجة محتويات الأداة والتفاعل مع المستخدم من ضمن كودك.

خاصية تجانب النص TextAlign

تعمل هذه الخاصية على إعداد او (تعود) بترصيف نص الأداة(تجانب نص الأداة) وقيمتها عضو من عداد الترصيف الأفقي HorizontalAlignment:الى اليسار Left او اليمين Right او Center الى المركز.فأداة صندوق النص لا تسمح لك بتنسيق النص(مزج خطوط او ألوان مختلفة) ولكن تستطيع ان تعمل على إعداد الخط الذي سيتم عرض النص به من خلال خاصية الخطFont, بالإضافة الى لون خلفية الأداة من خلال الخاصية BackColor.

خاصية تعدد الأسطر MultiLine

تحدد هذه الخاصية فيما اذا ستحتفظ اداة صندوق النص بنص ذو سطر مفرد او متعدد السطور. في كل مرة تضع فيها اداة صندوق نص على الفورم يتم تحجيمها من اجل نص بسطر مفرد وتستطيع في هذه الحالة تغير طول صندوق النص فقط(اتساعه) ولتغير هذا السلوك عليك وضع خاصية متعددة الأسطر الى صح . عندما تعمل على إنشاء صندوق نص متعدد الأسطر فانه يتوجب عليك وفي اغلب الحالات ان تعمل على إعداد واحدة او أكثر من الخاصيات التالية من خلال نافذة الخصائص: الطول الاعظمي MaxLength, أشرطة الانزلاق Scrollbars والتفاف النص Word Wrap.

خاصية الطول الاعظمي MaxLength

تحدد هذه الخاصية عدد الأحرف التي سيقبلها صندوق النص. فالقيمة الافتراضية لصندوق النص هي 32,767 والتي كانت العدد الاعظمي من الأحرف في إصدار الفيجوال بيسك6, فالعمل هنا على إعداد هذه الخاصية الى صفر 0 يمكن النص

من ان يحتوي على أي طول ,يمكن ان يصل الى 2,147,483,647 حرف, أما من اجل تحديد عدد الأحرف التي يستطيع ان يكتبها المستخدم ضع قيمة هذه الخاصية الى عدد الأحرف التي تريد ان لا يتجاوزها المستخدم .
ان اداة الطول الاعظمي لصندوق النص في اغلب الأحيان يتم العمل على تحديد قيمتها في تطبيقات إدخال البيانات,والتي تمنع المستخدم من إدخال حروف تتجاوز الحد الذي يستطيع حقل قاعدة البيانات ان يستوعبه, . فمثلا صندوق النص المحدد لإدخال أعداد الكتب القياسية الدولية(international standard book numbers (ISBNs)) لن تقبل أكثر من 13 حرف.

خاصية أشرطة الانزلاق Scrollbars

تمنح هذه الخاصية القدرة على تحديد أشرطة الانزلاق التي تريد ان تلحقها بأداة صندوق النص إذا ما تجاوز النص أبعاد exceeds dimensions لأداة ولكنك لا تستطيع ان تضع شريط انزلاق بالنسبة لصندوق نص ذو سطر مفرد حتى ولو تجاوز النص اتساع الأداة. أما صندوق النص المتعدد الأسطر يمكن ان يكون لديه شريط انزلاق أفقي horizontal او vertical عمودي,او كلاهما,فإذا ما ألحقت شريط انزلاق أفقيhorizontal بأداة صندوق النص فان النص لن يلتف بشكل آلي عندما يكتب المستخدم.فمن اجل البدء بخط جديد على المستخدم ان يضغط مفتاح الإدخال Enter ,هذه الترتيبات مفيدة في عمل محررات الكود في تلك الحالات التي يجب ان يتم تقسيم السطور بشكل واضح.
فإذا كان شريط الانزلاق الأفقي غير موجود فان الأداة تعمل على تقسيم السطر عندما يصل النص الى نهاية السطر وبالتالي فان النص يلتف بشكل أوتوماتيكي الى بداية سطر جديد,ولكن تستطيع تغير السلوك الافتراضي من خلال العمل على إعداد خاصية التفاف النص Word Wrap

خاصية التفاف النص WordWrap

تحدد هذه الخاصية فيما اذا سيلتف النص بشكل آلي عندما يصل الى حافة الأداة,وبشكل افتراضي قيمة هذه الخاصية هو صح. سنعمل فيما بعد مفكرة وتستطيع ان تختبر خاصية التفاف النص وأشرطة الانزلاق فيها.ولكن لاحظ هنا ان خاصية التفاف النص ليس لها أي تأثير فعلي على تقسيم السطور, فالسطور تلتف بشكل آلي ولا يوجد إشارة ارجاع (الشحطة المنخفض) عند نهاية السطر.

ملاحظة:

اداة صندوق النص ومن خلال خاصيتها "الطول الاعظمي " فعندما يتم العمل على إسناد القيمة صفر لها ، فان كلا من خاصية التفاف النص وتعدد الأسطر يتم وضعها الى فعال(صح) وشريط الانزلاق العمودي ايضا يصبح فعال.

خاصية قبول الرجوع ومفتاح التنقل AcceptsReturn, AcceptsTab

تحدد هاتين الخاصيتين كيفية تفاعل اداة صندوق النص مع مفتاح الرجوع(فتح سطر جديد Enter) ومفتاح التنقل Tab فمفتاح الإدخال Enterاو الرجوع(بالنسبة للملفات النصية) يعمل على تفعيل الزر الافتراضي على الفورم فإذا وجد زر فان الزر الافتراضي عادة يكون زر OK والذي من الممكن تفعيله بالمفتاح Enter,حتى ولو لم يكن لديه التَركيز.اما بالنسبة لأداة صندوق النص المتعددة الأسطر,ومهما يكن فنحن نريد ان نكون قادرين على استخدام مفتاح الإدخال لتغير الأسطر,عادة تكون القيمة الافتراضية للخاصية"قبول الرجوعAcceptsReturn " فعالة(صح)ولذا فان ضغط المفتاح Enter يعمل على إنشاء خط جديد على الأداة,فإذا ما عملت على إعداده الى خطأ (غير فعال) يبقى بإمكان المستخدمين إنشاء سطور جديدة في اداة صندوق النص,ولكن عليهم الضغط على CTRL+ENTER,أما اذا كانت الفورم لا تحوي على زر افتراضي فان مفتاح الإدخال يعمل على إنشاء سطر جديد بغض النظر عن اعدادات الخاصية AcceptsReturn. وبطريقة مماثلة تحدد الخاصية the AcceptsTab كيفية استجابة الأداة لمفتاح التنقل Tab , بشكل طبيعي يأخذك المفتاح Tabالى الأداة التالية وحسب ترتيب التنقل ,وبشكل عام نتجنب تغير الإعدادات الافتراضية لخاصية قبول التنقل AcceptsTab .من الممكن انك تريد من المفتاح Tabفي اداة صندوق النص المتعدد الأسطر ان يعمل على إدخال علامة المؤشرTab character في نص الأداة.لعمل هذا ضع خاصية "قبول التنقل "للأداة الى فعال (القيمة الافتراضية لها تكون غير فعال),فإذا غيرت القيمة الافتراضية يبقى باستطاعة المستخدمين التنقل الى الأداة التالية حسب ترتيب التنقل Tab بالضغط على CTRL+TAB,لاحظ ان خاصية AcceptsTab ليس لها تأثير على الادوات الأخرى,ولكن على المستخدمين ضغط CTRL+TAB للتنقل الى الأداة التالية,بينما يكون التركيز على اداة النص,ولكن باستطاعة المستخدمين استخدام مفتاح التنقل Tabللتحرك من أي اداة أخرى الى الأداة التالية على الفورم.

خاصية حالة الأحرف CharacterCasing

ليست بحاجة الى شرح فإنها تعمل على تحويل حالة الأحرف بين الحالة الكبيرة والصغيرة التي يعمل المستخدم على إدخالها فتستطيع من هذا الأداة تحويل مثلا جميع الأحرف التي يعمل على إدخالها المستخدم الى حالة الأحرف الصغيرة او الكبيرة حسب الحاجة.

خاصية محرف كلمة المرور PasswordChar

تعمل هذه الخاصية على تحويل الأحرف المكتوبة الى أي رمز او محرف تعمل أنت على تحديده,.فإذا كنت لا ترغب بعرض الحروف الحقيقة التي يكتبها المستخدم(عند إدخال كلمة المرور مثلا) استخدم هذه الخاصية لتحديد المحرف الذي تريده ان يظهر مكان كل حرف يعمل المستخدم على كتابته,القيمة الافتراضية لهذه الخاصية هي نص فارغ,والتي تخبر الأداة بعرض الحروف كما هي (عرض الأحرف الحقيقية) فإذا ما عملت على وضع رمز asteriskالنجمة(*) مثلا فان المستخدم سيرى النجمة مكان كل حرف يكتبه,مع الأخذ بعين الاعتبار ان هذه الخاصية لا تؤثر على النص الذي يكتبه المستخدم فهي مجرد قناع فقط لحماية النصوص الخاصة مثل كلمة المرور من ان يشاهد احد ما الذي تكتبه.

خاصية للقراءة فقط ,وخاصية التثبيت(الإغلاق) ReadOnly, Locked

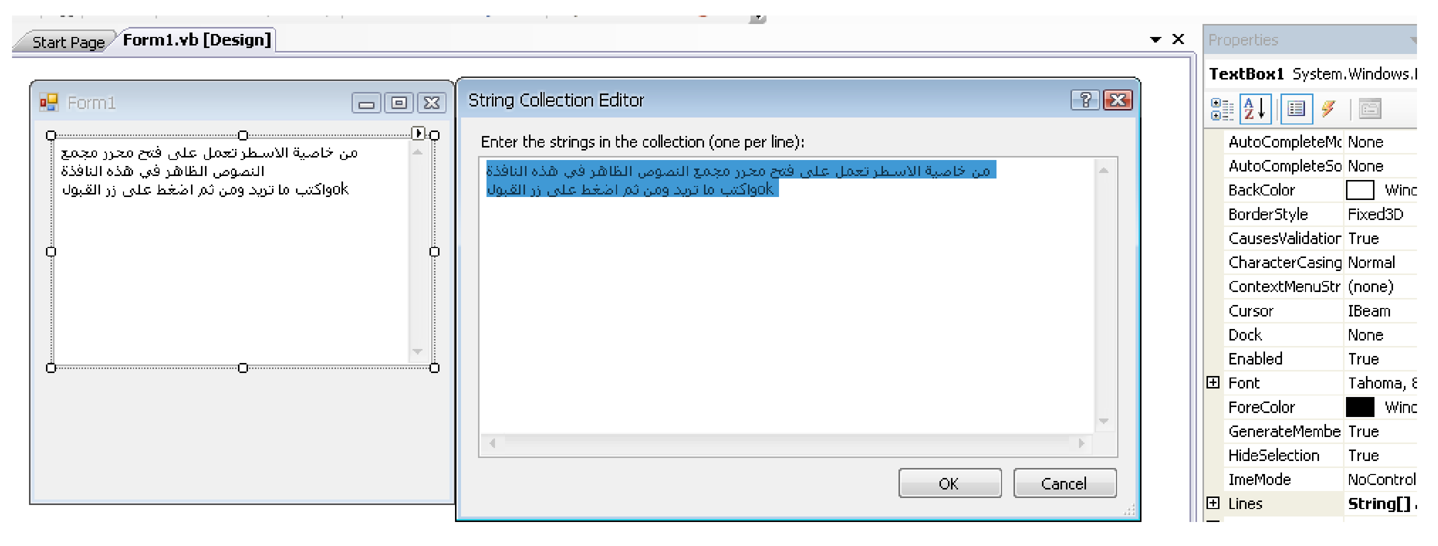
تعمل خاصية للقراءة فقطReadOnly على منع المستخدم من إدخال أي بيانات في الأداة التي يتم إعداد هذه الخاصية لها ,فهو يستقبل البيانات دون ان يستطيع ان يحررها,أما خاصية Locked التثبيت او الإغلاق فإنها تعمل على تثبيت الأداة في مكانها على الفورم وتستخدم هذه الخاصية وقت التصميم كي لا يتم نقل الأداة التي تم تنسيقها بالنسبة لباقي الادوات عن طريق الخطأ.

خاصيات معالجة النص Text-Manipulation Properties

معظم الخاصيات التي تعمل على معالجة النصوص لأداة صندوق النص متاحة وقت التنفيذ فقط,يقدم هذا المقطع تحليل breakdownلكل خاصية.

خاصية النص Text

الخاصية الأكثر أهمية لأداة صندوق النص هي خاصية النصText ,والتي تثبت نص الأداة,تستطيع العمل على إعداد هذه الخاصية وقت التصميم لعرض النص الأولي على الأداة.لاحظ انه يوجد طريقتين لإعداد هذه الخاصية وقت التصميم,فمن اجل صندوق نص وحيد السطر يتم وضع النص الى نص قصير,أما بالنسبة لصناديق النصوص المتعددة الأسطر افتح خاصية الأسطر Linesوأدخل النص في String Collection Editor window نافذة محرر جمع النصوص، والذي سيظهر في هذه النافذة كل فقرة تم إدخالها كسطر مفرد في النص,وعندما تنتهي اضغط OKلإغلاق النافذة,فالنص الذي أدخلته في نافذة محرر جمع النصوص سيتم وضعه على الأداة.وبالاعتماد على اتساع الأداة و اعدادات خاصية التفاف النص فيمكن للفقرة ان يتم تجزيئها الى عدة اسطر كما هو مبين في الشكل التالي:



أما وقت التنفيذ استخدم خاصية النص لاستخراج النص الذي تم إدخاله من قبل المستخدم او من اجل تبديل النص. خاصية النص هي بيانات نصية ويمكن ان تستخدم كمعامل نسي مع وظائف معالجة البيانات النصية للفيجوال بيسك.وتستطيع ان تعالجها manipulateايضا بواسطة مكونات فئة النصوص.التعبير التالي يعود بعدد الحروف في اداة صندوق النص:

```
Dim strLen As Integer = TextBox1.Text.Length
```

الطريقة IndexOf لفئة النصوص ستجد البيانات النصية المحددة في نص الأداة,فالعبارة التالية ترجع موضع occurrenceالمصادفة الأولى للنص "Visual "في النص:

```
Dim location As Integer
location = TextBox1.Text.IndexOf("Visual")
```

لتخزين محتويات الأداة في ملف استخدم عبارة كالتالية:

```
StrWriter.Write(TextBox1.Text)
```

وبشكل مشابه تستطيع أن تقرأ محتويات ملف نصي في أداة صندوق النص باستخدام عبارة مثل التالية:

```
TextBox1.Text = StrReader.ReadToEnd
```

حيث ان StrWriter و StrReader متغيرات مناسبة يتم التصريح عنهما ككاتب وقارئ البيانات على التسلسل StreamWriter , Stream Reader سترى لاحقا في الفصول القادمة كيفية طباعة الملفات النصية وكيفية الوصول إليها لا تشغل بالك كثيرا بها الآن فقط عليك التركيز على الخصائص وكيفية معالجتها في هذا الفصل.

لايجاد جميع النسخ من بيانات نصية ما في نص معين استخدم حلقة كالحلقة التي سنعرضها فيما يلي وهذه الحلقة تجد النسخ المتتالية للنص " Basic"ومن ثم تستمر بالبحث عن الحرف الذي يتبع النسخة السابقة من الكلمة في النص,فمن اجل إيجاد النسخة الأخيرة من بيانات النص في النص استخدم الطريقة LastIndexOf,. تستطيع كتابة حلقة مشابهة للمعروضة هنا لمسح النص بشكل تراجعي (بشكل راجع أي من النهاية الى البداية)

```
Dim startIndex = -1
        startIndex = TextBox1.Text.IndexOf("Basic", startIndex + 1)
        While startIndex > 0
            Debug.WriteLine("String found at " & startIndex)
            startIndex = TextBox1.Text.IndexOf("Basic", startIndex + 1)
End While
```

لتجرب مقطع الكود السابق ضع أداة TextBoxصندوق نص متعدد الأسطر وأداة Button زر على الفورم ومن ثم ضع عبارات الكود السابق في معالج حدث النقر على الزر,شغل التطبيق ومن ثم ادخل نص في اداة صندوق النص وتأكد من ان النص الذي أدخلته يحتوي على الكلمة Basic او غيّر الكود ليعمل بحث عن كلمة أخرى ومن ثم انقر على الزر.لاحظ ان الطريقة IndexOf تنجز بحث في حالة الحساسية للأحرف أي ان هذه الطريقة حساسة لحالة الأحرف سواء كانت صغيرة او كبيرة.

استخدم الطريقة لتبديل نص بنص أخر ضمن السطر. والطريقة لتقسيم السطر الى مكونات اصغر (مثل الكلمات) و بإمكانك استخدام أي طريقة أخرى يتم عرضها بواسطة فئة النصوص لمعالجة نص الأداة, فالعبارات التالية تعمل على إلحاق نص ما بنص موجود سابقا على الأداة:

```
TextBox1.Text = TextBox1.Text & newString
```

هذه العبارة تظهر فقط في تطبيق فيجوال بيسك6 الذي يعمل على معالجة نص اداة صندوق النص وهي غير فعالة لإلحاق نص بالأداة, وخاصة, اذا كانت الأداة تحتوي مسبقا على الكثير من النصوص,.تستطيع الآن ان تستخدم الطريقة AppendText لتنزّيل (او إلحاق) نصوص الى الأداة والتي هي فعالة أكثر بكثير من الطريقة السابقة و التي تعالج خاصية النص للأداة بشكل مباشر,فمن اجل إلحاق نص الى اداة صندوق النص استخدم العبارة التالية:

```
textBox1.AppendText(newString)
```

الطريقة AppendText تعمل على إلحاق نص محدد الى الأداة كما هو بدون ان تقسم الأسطر بين الاستدعاءات المتلاحقة. فإذا كنت تريد ان تلحق فقرة مستقلة بنص الأداة عليك إدخال مقسم الأسطر بشكل صريح كما في التالي(حيث ان vbCrLf هو ثابت من اجل عودة المؤشر/ وتوليد سطر جديد)

```
TextBox1.AppendText(newString & vbCrLf)
```

خاصية الأسطر Lines

بالإضافة الى خاصية النص تستطيع الوصول الى النص على الأداة باستخدام خاصية Linesالأسطر,وخاصية الأسطر هي مصفوفة نصية,وكل عنصر يثبت paragraph فقرة من النص. فالفقرة الأولى يتم تخزينها في العنصر"Lines(0) سطر(0) " والفقرة الثانية يتم تخزينها في "Lines(1) سطر(1)" وهكذا . تستطيع ان تعمل دوران على السطور باستخدام حلقة كالتالية:

```
Dim iLine As Integer
        For iLine = 0 To TextBox1.Lines.GetUpperBound(0)
          { process string TextBox1.Lines(iLine) }
Next
```

يتوجب عليك تبديل السطر الذي بين الأقواس المنحنية بالكود المناسب, لان خاصية الأسطر هي مصفوفة فإنها تدعم الطريقة GetUpperBound " احصل على الحد الأعلى " و التي تعود بفهرس العنصر الأخير للمصفوفة , كل عنصر من

مصفوفة الأسطر هو نص ,وتستطيع استدعاء أيا من طرق فئة النصوص لمعالجته, عليك ان تتذكر دائما انك لا تستطيع ان تحول النصوص التي على الأداة من خلال تحرير مصفوفة الأسطر, لكن ومهما يكن تستطيع ان تعمل على إعداد نص الأداة بإسناد مصفوفة نصية الى خاصية Linesالأسطر .

خاصيات اختيار النص Text-Selection Properties

تزودك اداة صندوق النص بثلاث خاصيات لمعالجة النص المختار بواسطة المستخدم وهي :SelectedText النص المختار و SelectionStart بداية الاختيار و SelectionLength طول الاختيار. يستطيع المستخدم ان يختار مجال النص بواسطة click-and-drag operation العملية نقر- سحب وبالتالي فان النص المختار سيظهر بلون معكوس, تستطيع التمكن من الوصول الى النص المختار من ضمن كودك من خلال الخاصية SelectedText"النص المختار" وتحدد موضعها في نص الأداة من خلال الخاصية SelectionStart والخاصية SelectionLength

النص المختار SelectedText

هذه الخاصية تعود بالنص الذي تم اختياره وتمنحك القدرة على معالجة الاختيار الحالي من ضمن كودك. فمثلا تستطيع ان تبدل النص الذي تم اختياره بإسناد قيمة جديدة الى خاصية "النص المختار SelectedText" . لتحويل النص المختار الى نص جديد تكون فيه حالة الأحرف كبيرة، استخدم الطريقة ToUpperالأعلى من فئة النصوص String class.

```
TextBox1.SelectedText = TextBox1.SelectedText.ToUpper
```

بداية وطول الاختيار SelectionStart, SelectionLength

استخدم هاتين الخاصتين لقراءة النص المختار من قبل المستخدم على الأداة ,او لاختيار نص من ضمن كودك , خاصية بداية الاختيار SelectionStart تعود او تضع موقع الحرف الأول من النص المختار بطريقة مشابهة لوضع مشيرة الكتابة cursor عند موضع معين في النص واختيار النص بسحب الفارة. أما خاصية SelectionLength طول الاختيار فإنها تعود او تضع طول النص المختار.

على فرض ان المستخدم يبحث عن كلمة "فيجوال " في نص الأداة تعمل الطريقة IndexOf على إيجاد النص ولكنها لا تختاره,العبارات التالية تختار الكلمة في النص وتعمل على إبرازها وتجلبها الى شاشة العرض, لذا فان المستخدم يستطيع يعلم عليها بشكل مباشر:

```
Dim seekString As String = "فيجوال"
        Dim strLocation As Long
        strLocation = TextBox1.Text.IndexOf(seekString)
        If strLocation > 0 Then
            TextBox1.SelectionStart = strLocation
            TextBox1.SelectionLength = seekString.Length
        End If
TextBox1.ScrollToCaret()
```

تدعم اداة صندوق النص الطريقة ScrollToCaret والتي تجلب مقطع النص الذي يتم البحث عنه الى الواجهة

ملاحظة: موضع المشرة في الأداة Locating the Cursor Position in the Control
ان الخاصية SelectionStartوالخاصية SelectionLengthدائما يكون لكل منهما قيمة حتى ولو لم يتم اختيار أي نص على الأداة. في هذه الحالة تكون خاصية SelectionLengthصفر, وخاصية SelectionStart هي الموضع الحالي للمؤشر في النص,فإذا أردت ان تدخل بعض النص عند موضع مؤشر,ببساطة اعمل على إسناده الى خاصية SelectedText, حتى ولو لم يتم اختيار نص على الأداة.

إخفاء الاختيار HideSelection

النص المختار في صندوق النص لا يبقى بارز عندما ينتقل المستخدم الى اداة أخرى او الى الفورم,لتغير هذا السلوك الافتراضي , اعمل على وضع الخاصية HideSelectionالى False" غير فعال " . ان استخدام هذه الخاصية هو لحفظ النص المختار بشكل بارز حتى ولو انتقل التركيز الى فورم أخرى او الى صندوق حوار مثلا "إيجاد واستبدال" أما اذا ما أبقيت القيمة الافتراضية والتي هي True "فعال" فهذا يعني ان النص لم يبقى بارزا عندما يفقد صندوق النص التركيز.

طرق اختيار النص Text-Selection Methods

بالإضافة الى الخاصيات,فان لأداة صندوق النص طرق من اجل اختيار النص, تستطيع ان تختار بعض النصوص باستخدام الطريقة "اختر Select" والتي لها الشكل العام التالي:

```
TextBox1.Select(start, length)
```

الطريقة "اختر" مكافئة لوضع الخاصيتان " SelectionStart بداية الاختيار " SelectionLengthوطول الاختيار" فمن اجل اختيار الحروف من 100 الى 105 على الأداة, استدعي الطريقة " Select اختر" ومرر لها القيم 99 و6 كمعاملات نسبية arguments .

```
TextBox1.Select(99, 6)
```

كتذكير فان ترتيب الحروف يبدأ من الصفر(فهرس الحرف الأول هو الصفر والحرف الثاني هو الواحد والحرف الثالث هو 2 وهكذا أما فهرس الحرف الأخير فيكون طول النص منقوصا منه واحد).اذا كان مجال الأحرف التي عملت على اختيارها تحتوي على hard line breaks رموز تقطيع الأسطر يجب ان تأخذها بعين الاعتبار,وكل رمز تقطيع للأسطر يتم احتسابه كحرفين(رجوع المؤشر و تحميل السطر carriage return and line feed) . اذا كانت اداة صندوق النص تحتوي على النص ABCDEFGHI فان العبارة التالية ستعمل على اختيار المجالDEFG

```
TextBox1.Select(3, 4)
```

فإذا عملت على إدخال تقطيع السطر بين كل ثلاث أحرف فان النص سيصبح كما يلي(ونفس العبارة السابقة ستعمل على اختيار الأحرف DE):

```
ABC
DEF
GHI
```

في الواقع لقد تم ايضا اختيار الرموز الخاصة التي تفصل السطرين الأول والثاني,ولكن الرموز الخاصة لا يتم عرضها وبالتالي فانه لا يمكن ان يتم إبرازها. بقي طول الاختيار في العبارة السابقة هو 4 ولكن كما قلنا الحروف او الرموز الخاصة لا يتم عرضها.يوجد تنوع لطريقة Select"اختر" وهو الطريقة " Select All اختر الجميع" والي تختار كل النص الذي على الأداة.

التراجع عن التحريرات Undoing Edits

من الميزات المهمة لأداة صندوق النص هي انه من الممكن وبشكل آلي التراجع عن معظم عملية التحرير والتحديث الحديثة.فلتراجع عن عملية من ضمن كودك يجب ان تتفحص قيمة الخاصية CanUndo "يمكن التراجع" فإذا كانت True صح عندها يمكن للأداة التراجع عن العملية,ومن ثم تستطيع الطريقة undo "تراجع" للتراجع عن معظم التحديثات الحديثة.

عملية التحرير هي إدخال او حذف حروف.فإدخال نص بدون أي حذف تعتبر عملية مفردة,وسوف يتم التراجع بخطوة وحيدة ايضا,حتى ولو استغرق المستخدم ساعات في إدخال النص(ولكن بدون عمل أي تصحيح)تستطيع جعل كل النص تختفي باستدعاء وحيد لطريقة Undo method "التراجع",ولكن ولحسن الحظ فان عملية الحذف ستصبح العملية الأحدث

والتي يمكن التراجع عنها باستدعاء أخر لطريقة Undo"تراجع". في الواقع طريقة "التراجع" هي toggle عقدة تمفصل أي(عمل/التراجع عن العمل)فعند استدعاءها لأول مرة فإنها تعمل على التراجع عن عملية التحرير الأخيرة وإذا ما عملت على استدعاءها مرة أخرى فإنها تعكس العملية أي redoesتعيد عمل العملية الأخيرة.عملية حذف نص يمكن التراجع عنها فقط اذا لم تحدث عملية تحرير في الوقت المعني.تستطيع تعطيل عملية redo "عكس التراجع(اعادة عمل) " باستدعاء الطريقة ClearUndo"تنظيف التراجع"والتي تنظف undo buffer منطقة التراجع للأداة.وستعمل على استدعاء هذه الطريقة من ضمن معالج حدث الأمر تراجع لمنع من ان تصبح عملية التراجع عكس التراجع redo.في معظم الحالات ستمنح المستخدم خيار عملية عكس التراجع redoو اعادة عمل وخاصة كون عملية التراجع يمكن ان تعمل على حذف كمية هائلة من النص للأداة.

تدريب

الآن بعد ان انتهينا من اداة صندوق النص إليك التدريب التالي:

مشروع المفكرة The TextPad Project

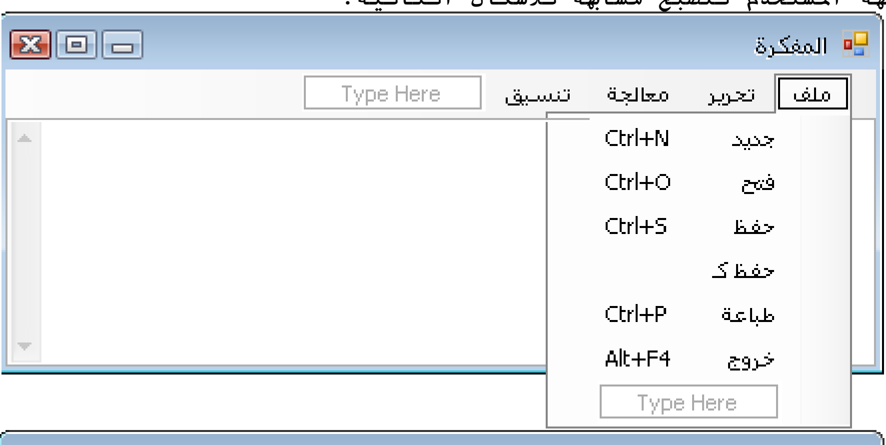
تطبيق المفكرة يوضح معظم خاصيات اداة صندوق النص وطرقها التي تم شرحها حتى الآن.فالمفكرة هي محرر نصوص أساسي تستطيع ان تضمنه في برامجك,وتخصصه من اجل تطبيقات محددة.

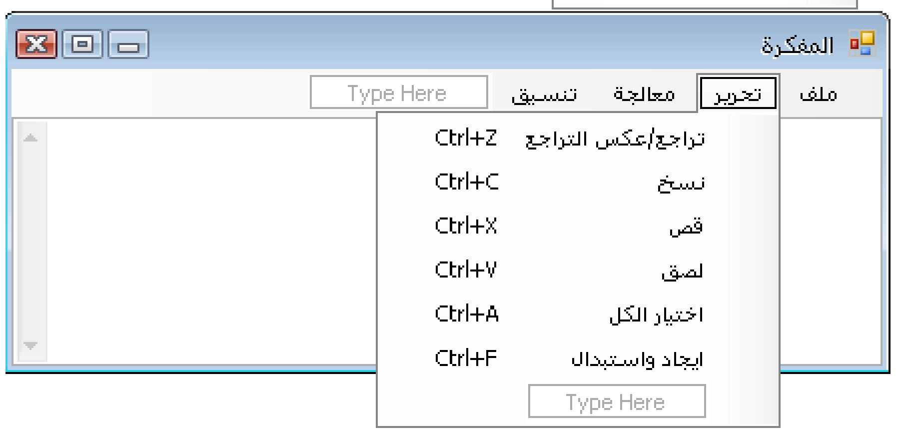
افتح مشروع جديد وسمه "قارئة النصوص" وعند فتح الفورم اعد تسمية الفورم من خلال خاصية النص Text الى "المفكرة" ضع اداة قائمة MenuStripعلى الفورم ورصفها dockالى أعلى الفورم,ومن ثم ضع اداة صندوق نص على الفورم،وضع في خاصية الاسم البرمجي name txtEditor " ومن ثم اعمل على إعداد الخاصيات التالية: Multiline to True خاصية تعدد الأسطر الى صح،MaxLength to 0 وخاصية الطول الاعظمي الى صفر(لتحرير مستندات النصوص وبأي طول) ،HideSelection to False وخاصية إخفاء الاختيار الى خطأ(حيث ان النص المختار سيبقى بارزا عندما لا يكون التركيز على الفورم الرئيسي)،Dock to Fill واخيرا خاصية الترصيف الى ملء(مما يؤدي الى ان اداة صندوق النص ستملئ الفورم).

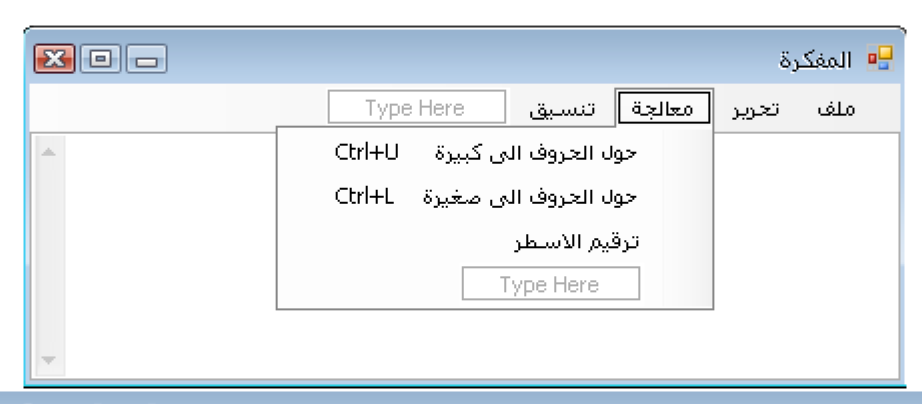
شريط MenuStrip القوائم للفورم يحتوي على جميع الأوامر التي تتوقع إيجادها في تطبيق محرر نصوص وتم جدولتها في الجدول التالي:

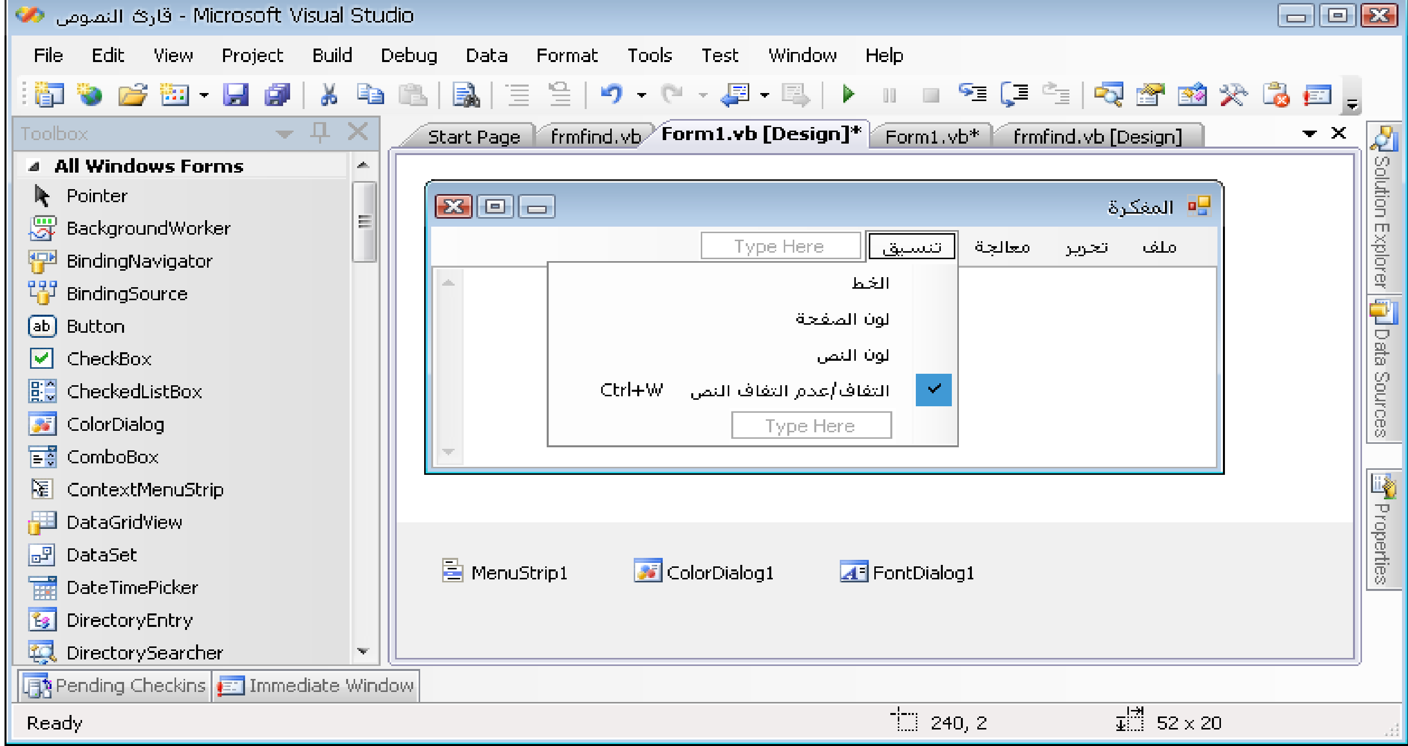
القائمة Menu	الأمر Command		الوصف Description
ملف File	New	جديد	يزيل النص من اجل نص جديد
	Open	فتح	يحمل ملف نصي جديد من القرص
	Save	حفظ	يحفظ النص في ملفه على القرص
	Save As	حفظ كـ	يحفظ النص باسم جديد للملف على القرص
	Print	طباعة	لطباعة النص
	Exit	خروج	لإنهاء التطبيق
تحرير Edit	Undo/Redo	تراجع/عكس التراجع	من اجل التراجع عن عملية /وإعادة عمل ما تم التراجع عنه
	Copy	نسخ	نسخ النص المختار الى الحافظة
	Cut	قص	قص النص المختار
	Paste	لصق	لصق محتوى الحافظة الى المحرر
	Select All	اختر الكل	اختيار كل النص الذي في الأداة
	Find & Replace	إيجاد واستبدال	عرض صندوق حوار مع خيار إيجاد واستبدال
معالجة Process	Convert To Upper	حول الى الكبير	تحويل النص المختار الى حالة الأحرف الكبيرة
	Convert To Lower	حول الى الصغير	تحويل النص المختار الى حالة الأحرف الصغيرة
	Number Lines	رقم الأسطر	ترقيم سطور النص
تنسيق Format	Font	الخط	وضع خط النص مثل الحجم والموصفات الأخرى
	Page Color	لون الصفحة	وضع لون الصفحة من خلال لون خلفية الأداة
	Text Color	لون النص	وضع لون النص
	WordWrap	التفاف النص	بند متفصل والذي يحول التفاف النص الى ON او OFF

اعمل على إعداد واجهة المستخدم لتصبح مشابهة للأشكال التالية:



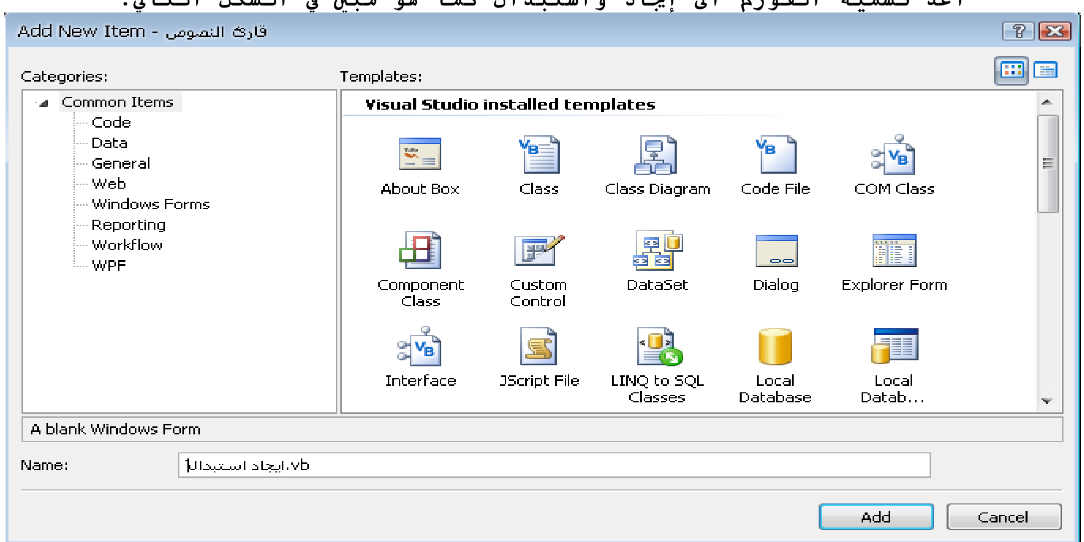






ضع الادوات التالية على الفورم:اداة حوار الخط FontDialog1 وأداة حوار اللون:ColorDialog1 بالاضافة الى اداة القوائمMenuStrip1 وهذه الادوات الثلاث ستظهر في صينية محرر الكود أي أسفل الفورم كما هو مبين في الشكل الأعلى ,بالنسبة لأداة القوائم ضع فيها القوائم الموضحة في الاشكال السابقة والجدول السابق أما بالنسبة لمفاتيح الاختصار تستطيع اعدادها كما هي موضحة في الاشكال من خلال خاصية shortcutkeyفي نافذة الخصائص حيث تحدد كل واحدة من القوائم الفرعية ومن ثم ومن خلال هذه الخاصية تعمل على تحديد المفتاح الذي تراه مناسبا مع احد المفاتيح الرئيسية إما ctrl اوalt اوshift .هذا بالإضافة الى ضرورة جعل الخاصيات التالية الإضافية للفورمات التي تكتب فيها باللغة العربية:خاصيةrighttolift=yes

وخاصية righttoliftlayout=true وذلك ليكون تخطيط الفورم من اليمين الى اليسار والكتابة ايضا تبدأ من اليمين الى اليسار وكما هو موضح في الاشكال السابقة حيث ان تخطيط الفورم من اليمين الى اليسار والنصوص واسماء الادوات كلها باللغة العربية.بقية خطوة أخرى عليك عملها ألا وهي إضافة فورم جديدة الى المشروع من قائمة مشروع project اختار إضافة فورم ويندوز add windows form ومن صندوق حوار إضافة بند الذي سيظهر لك اعد تسمية الفورم الى إيجاد واستبدال كما هو مبين في الشكل التالي:



الآن وبعد أن عملت على إضافة فورم ثانية الى المشروع من خاصية الاسم البرمجي name في نافذة الخصائص اعد تسمية الفورم الثانية الى frmfindوضع عليها أربعة أدوات زر واداتي عنوان وصندوقي نص وأداة صندوق اختيار checkbox اعد تسمية هذه الادوات من خلال خاصية النص الى ما يبينه الشكل التالي:

ايجاد واستبدال

البحث عن

ايجاد

ايجاد التالي

تبديل بـ

تبديل

تبديل الكل

تحسس حالة الاحرف

إذا كانت بعض الامور غيرمفهومة بالنسبة لك فلا تقلق ساعمل هنا على شرح الكود وتبقى بعض الامور سيتم شرحها في وقتها ان شاء الله.

إليك جدول القوائم واسمائها البرمجية (name) التي سأستخدمها هنا:

الادوات واسماء الـ text (خاصية النص)	الاسم البرمجي name المستخدم في الكود
ملف(قائمة رئيسية)	file
جديد(قائمة فرعية من قائمة ملف)	filenew
فتح(قائمة فرعية من قائمة ملف)	fileopen
حفظ(قائمة فرعية من قائمة ملف)	filesave
حفظ كـ(قائمة فرعية من قائمة ملف)	fileas
طباعة(قائمة فرعية من قائمة ملف)	filep
خروج(قائمة فرعية من قائمة ملف)	fileexit
تحرير(قائمة رئيسية)	edit
تراجع/عكس التراجع(قائمة فرعية من تحرير)	editundo
نسخ(قائمة فرعية من قائمة تحرير)	editcopy
قص(قائمة فرعية من قائمة تحرير)	editcut
لصق(قائمة فرعية من قائمة تحرير)	editpaste
اختيار الكل(قائمة فرعية من قائمة تحرير)	editall
إيجاد استبدال(قائمة فرعية من قائمة تحرير)	editfind
معالجة(قائمة رئيسية)	process
حول الحروف الى كبيرة(فرعية من قائمة معالجة)	processup
حول الحروف الى صغيرة(فرعية من قائمة معالجة)	processlow
ترقيم الأسطر(قائمة فرعية من قائمة معالجة)	processnumber
تنسيق(قائمة رئيسية)	format
الخط(قائمة فرعية من قائمة تنسيق)	formatfont
لون الصفحة(قائمة فرعية من قائمة تنسيق)	formatcolorpage
لون النص(قائمة فرعية من قائمة تنسيق)	formatcolortxt
التفاف/عدم الالتفاف(قائمة فرعية من قائمة تنسيق)	formatwordwrap
المفكرة(الفورم الرئيسية)	frmTextPad
اداة صندوق النص(النص text اتركه فارغ)	txteditor
ايجاد واستبدال(الفورم الثانوية المضافة الى المشروع)	frmfind
البحث عن(اداة عنوان على الفورم الثانوي)	Label1
تبديل بـ(اداة عنوان على الفورم الثانوي)	Label2
اداة صندوق نص(على الفورم الثانويةدع النص فارغ)	txtfind
اداة صندوق نص(على الفورم الثانويةدع النص فارغ)	txtreplac
إيجاد(اداة زر على الفورم الثانوية)	btnfind
إيجاد التالي(اداة زر على الفورم الثانوية)	tnfindnext
تبديل(اداة زر على الفورم الثانوية)	btnreplace
تبديل الكل(اداة زر على الفورم الثانوية)	btnreplaceall
تحسس حالة الأحرف(صندوق اختيار على الفورم الثانوية)	Checkcasesensitive

هذه جميع الادوات مع خاصية النص والاسم البرمجي لكل منها مع القوائم التي قمت بعملها مع قوائمها الفرعية بقي انوه ان عليك جعل خاصيةحالةالاختيارchecked = checkstate للقائمة الفرعية التفاف/عدم الالتفاف(قائمة فرعية من قائمة تنسيق)

بقي عليك ان تنسق الادوات والقوائم كما هي مبينة في الاشكال السابقة وهي عملية عادية جدا

```vb
Public Class frmTextPad
    Private Sub filenew_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
filenew.Click
        txteditor.Clear() كود امر جديد من قائمة تحرير
End Sub
    Private Sub editcut_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
editcut.Click
        Clipboard.SetText(txteditor.SelectedText) كود امر القص من قائمة تحرير
        txteditor.SelectedText = ""
    End Sub
    Private Sub editpaste_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
editpaste.Click
        If Clipboard.ContainsText Then
            txteditor.SelectedText = Clipboard.GetText كود امر اللصق من قائمة تحرير
        End If
    End Sub
    Private Sub editfind_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
editfind.Click
```

```
        frmfind.Show() كود امر ايجادو استبدال من قائمة تحرير والذي يعمل على استدعاء الفورم الثانوية
    End Sub
    Private Sub editcopy_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
editcopy.Click
        If txteditor.SelectionLength > 0 Then
            Clipboard.SetText(txteditor.SelectedText) كود امر النسخ من قائمة تحرير
        End If
    End Sub
    Private Sub processnumber_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
processnumber.Click
        Dim iLine As Integer
        Dim newText As New System.Text.StringBuilder() كود امر ترقيم الاسطر من قائمة معالجة
        For iLine = 0 To txteditor.Lines.Length - 1
            newText.Append((iLine + 1).ToString & vbTab & _
                           txteditor.Lines(iLine) & vbCrLf)
        Next
        txteditor.SelectAll()
        Clipboard.SetText(newText.ToString)
        txteditor.Paste()
    End Sub
    Private Sub processlow_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
processlow.Click
        txteditor.SelectedText = txteditor.SelectedText.ToLower امر تحويل الأحرف الى صغير من قائمة معالجة
    End Sub
    Private Sub processup_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
processup.Click
        txteditor.SelectedText = txteditor.SelectedText.ToUpper امرتحويل الأحرف الى كبيرة من قائمة معالجة
    End Sub
    Private Sub editundo_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
editundo.Click
        If editundo.Text = "Undo" Then
            If txteditor.CanUndo Then
                txteditor.Undo() كود امر التراجع/عكس التراجع من قائمة تحرير
                editundo.Text = "Redo"
            End If
        Else
            If txteditor.CanUndo Then
                txteditor.Undo()
                editundo.Text = "Undo"
            End If
        End If
    End Sub
    Private Sub txteditor_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs)
Handles txteditor.KeyPress
        Dim ch As Char
        If System.Char.IsControl(e.KeyChar) Then Exit Sub
        ch = Char.ToUpper(e.KeyChar)
        Select Case ch.ToString
            Case "@"
                txteditor.SelectedText = "AT"
            Case "#"
                txteditor.SelectedText = "BPT"
            Case "$"
                txteditor.SelectedText = "DLR"
            Case "%"
                txteditor.SelectedText = "O/O"
            Case "&"
                txteditor.SelectedText = "AND"
            Case Else
                txteditor.SelectedText = ch
        End Select
        e.Handled = True
    End Sub
    Private Sub txteditor_KeyUp(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyEventArgs) Handles
txteditor.KeyUp
        Select Case e.KeyCode
            Case Keys.F5 : txteditor.SelectedText = Now().ToLongDateString
            Case Keys.F6 : txteditor.SelectedText = Now().ToLongTimeString
            Case Keys.F7 : txteditor.SelectedText = "MicroWeb Designs, Inc."
            Case Keys.F8 : txteditor.SelectedText = "Another user-supplied string"
        End Select
    End Sub
    Private Sub txteditor_TextChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles
txteditor.TextChanged
        editundo.Text = "Undo"
    End Sub
    Private Sub formatwordwrap_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
formatwordwrap.Click
        txteditor.WordWrap = formatwordwrap.Checked  كود امرالتفاف النص من قائمةتنسيق
    End Sub
    Private Sub formatcolortxt_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
formatcolortxt.Click
        ColorDialog1.Color = txteditor.BackColor
        If ColorDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            txteditor.ForeColor = ColorDialog1.Color
```

```
        End If

    End Sub
    Private Sub formatcolorpage_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
formatcolorpage.Click
        ColorDialog1.Color = txteditor.ForeColor
        If ColorDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            txteditor.ForeColor = ColorDialog1.Color
        End If
    End Sub
    Private Sub formatfont_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
formatfont.Click
        FontDialog1.ShowApply = True
        If FontDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            txteditor.Font = FontDialog1.Font
        End If
    End Sub
End Class
```

مناقشة الكود

أوامر التحرير The Editing Commands

الخيارات المتاحة في قائمة تحرير تعمل على تبادل النص من والى الحافظة.فمن اجل تطبيق المفكرة كل ما تحتاج معرفته حول الحافظة هو الطريقة SetText "ضع نص" والتي تعمل على وضع نص تم اختياره حاليا في الحافظة,والطريقة GetText "احصل على النص" والتي تستخلص المعلومات من الحافظة
فالأمر نسخ مثلا,يتم تنفيذه من خلال سطر وحيد من الكود(حيث ان txtEditor هو الاسم البرمجي لأداة صندوق النص) ونفس الشيء بالنسبة للأمر قص والذي يعمل ايضا على إزالة النص المختار والذي تم قصه(راجع الكود في القائمة السابقة)
اذا ما احتوت الحافظة على تنسيق أخر غير النص (مثل صورة)من تم وضعها من تطبيق أخر او أي نوع أخر من البيانات فان اداة صندوق النص لا تستطيع معالجته وعملية اللصق ستفشل وهذا السبب الذي أدى بان نضع الشرط If وبإمكانك من خلال إدخال العبارة Else ان تقدم للمستخدم معلومات بان البيانات في الحافظة Clipboard لا يمكن استخدامها مع تطبيق خاص بتحرير النصوص.

أوامر قوائم المعالجة والتنسيق The Process and Format Menus

ان هذه الأوامر بسيطة,فأوامر قائمة التنسيق تفتح صناديق حوار اللون والخط وتعمل على تغير Font الخط,ولونه ForeColor, بالإضافة الى خاصية لون الخلفية BackColor.وسوف تتعلم كيف تستخدم هذه الادوات في الفصل القادم ان شاء الله,. الأوامر تحويل الأحرف الى صغيرة وكبيرة تعمل على إسناد خاصية تحويل النص المختار SelectedText للأداة الى الحالة الموافقة (كبير او صغير)(راجع كود أمر تحويل حالة الأحرف الى كبير او صغير)
لاحظ ان الكود في أمر تحويل حالة الأحرف يستخدم الخاصية SelectedText لتحويل فقط النص الذي تم اختياره,وليس كامل المستند. الأمر ترقيم الأسطر يعمل على إدخال أرقام الأسطر أمام كل سطر من النص ويوضح كيفية معالجة كل سطر من النص بشكل مستقل ,ولكنه لا يعمل على رقم الأسطر ولم نعمل على عمل آلية لمنع المستخدم من تحديث ترقيم الأسطر او إدخال/حذف الترقيم للأسطر بعد ان يتم ترقيمها مرة,فاستخدم هذه الميزة لترقيم الأسطر فقط قبل ان تعمل على حفظ النص.لاحظ ان معالج حدث ترقيم الأسطر يستخدم المتغير من نوع الكائن StringBuilder،حيث ان الفئة StringBuilder class "باني النصوص" مكافئة لفئة النصوص String class وتعرض نفس الطرق والخصائص,ولكنها أسرع في معالجة النصوص الديناميكية من فئة النصوص .

العمليات بحث واستبدال(او إيجاد واستبدال) Search and Replace Operations

الخيار الأخير و الأكثر أهمية في قائمة تحرير يعرض صندوق حوار "إيجاد واستبدال"(او يعرض الفورم الثانوية التي عملنا على إضافتها الى المشروع)وصندوق الحوار هذا يعمل بطريقة مماثلة لصناديق الحوار التي تراها في التطبيقات الأخر مثل الورد او أي تطبيق ويندوز أخر. الأزرار في صندوق الحوار"إيجاد واستبدال" تفسر نفسها تقريبا self-explanatory .
الأمر"إيجاد" يعمل على إيجاد النسخة الأولى من النص المحدد في النص بعد موضع المشرة.وإذا ما تم إيجاد أكثر من نسخة فان الأزرار "إيجاد التالي" و"تبديل بـــ" و"تبديل الكل" سيتم تفعيلها.
اعمل على تصميم الفورم الثانية كما تظهره الأشكال السابقة,واعمل على إعداد الخاصية TopMost الى صح فنحن نريد من الفورم الثانية ان تبقى أمام الفورم الرئيسية,حتى ولو لم يكن التركيز عليها.بالاعتماد على صندوق الاختبار لحالة الأحرف فانه سيتم تحديد نوعية البحث هل سيتم في حالة Case Sensitive الحساسية للأحرف أم لا :لاحظ صندوق الاختبار الذي عملنا على إضافته في أسفل الزاوية اليمينية وتم تسميته من خلال خاصية النص الى" تحسس حالة الأحرف".
في فورم او صندوق حوار "إيجاد واستبدال" اذا ما تم إيجاد النص المنشود في نص اداة صندوق النص للفورم الرئيسية فان البرنامج يعمل على إبرازه من خلال اختياره,بالإضافة الى ان البرنامج يعمل على استدعاء طريقة صندوق النص ScrollToCaret "انزلق الى العلامة" لجلب الاختيار الى شاشة العرض.وزر "إيجاد التالي"يأخذ بعين الاعتبار موضع المؤشر ويبحث عن المطابقة بعد الموضع الحالي.فإذا ما حرك المستخدم المؤشر الى مكان أخر ومن ثم ضغط زر"إيجاد التالي"فان البرنامج سيجد النسخة الأولى من النص بعد الموقع الحالي للمؤشر pointer ,وليس بعد المطابقة الأخيرة وبالطبع تستطيع دائما ان تحتفظ بمسار الإيجاد بالنسبة لكل عملية مطابقة ومن ثم تستمر بالبحث من هذا الموضع . إليك كود الفورم الثانية كاملاً :

```
Public Class frmfind
'زر تبديل الكل ينجز البحث مع التحسس لحالة الأحرف
' فإذا كنت تريده ان يعمل بغض النظر عن حالة الأحرف فعليك البحث عن كل نسخة وتبديلها
' من ضمن كودك بدلا من استخدام طريقة InStr() لكلمة البحث باستخدام الوظيفة
' String class التابعة لفئة النصوص Replace التبديل
    Private Sub btnfind_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnfind.Click
        Dim selStart As Integer
        If Checkcasesensitive.Checked = True Then
            selStart = frmTextPad.txteditor.Text.IndexOf(txtfind.Text, StringComparison.Ordinal)
        Else
            selStart = frmTextPad.txteditor.Text.IndexOf(txtfind.Text,
StringComparison.OrdinalIgnoreCase)
        End If
        If selStart = -1 Then
```

```
                MsgBox("Can't find word")
                Exit Sub
            End If
            frmTextPad.txteditor.Select(selStart, txtfind.Text.Length)
'اذا تم إيجاد تطابق،سيتم تفعيل الأزرار الاخرى
            btnfindnext.Enabled = True
            btnreplace.Enabled = True
            btnreplaceall.Enabled = True
'تاكد من ان النص الذي تم اختيارة ظاهر(او مرئي)
            frmTextPad.txteditor.ScrollToCaret()
        End Sub
        Private Sub btnfindnext_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnfindnext.Click
            Dim selStart As Integer
            Dim srchMode As Microsoft.VisualBasic.CompareMethod
            If Checkcasesensitive.Checked = True Then
                srchMode = Microsoft.VisualBasic.CompareMethod.Binary
            Else
                srchMode = Microsoft.VisualBasic.CompareMethod.Text
            End If
'بداية البحث بعد الاختيار الحالي فنحن فرضنا ان الاختيار الحالي هو الموضع السابق للكلمة
            If Checkcasesensitive.Checked = True Then
                selStart = frmTextPad.txteditor.Text.IndexOf(txtfind.Text,
frmTextPad.txteditor.SelectionStart + 1, StringComparison.Ordinal)
            Else
                selStart = frmTextPad.txteditor.Text.IndexOf(txtfind.Text,
frmTextPad.txteditor.SelectionStart + 1, StringComparison.OrdinalIgnoreCase)
            End If
            If selStart = -1 Then
                MsgBox("No more matches")
                Exit Sub
            End If
            frmTextPad.txteditor.Select(selStart, txtfind.Text.Length)
            frmTextPad.txteditor.ScrollToCaret()
        End Sub
        Private Sub btnreplace_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnreplace.Click
            'تبديل الاختيار الحالي
            If frmTextPad.txteditor.SelectedText <> "" Then
                frmTextPad.txteditor.SelectedText = txtreplace.Text
            End If
'ومن ثم إيجاد النسخة الثانية من كلمة البحث
            btnfindnext_Click(sender, e)
        End Sub
        Private Sub btnreplaceall_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnreplaceall.Click
            Dim curPos, curSel As Integer
'حفظ مسار الاختيار الحالي
'وتخزينه بعد عملية التبديل
            curPos = frmTextPad.txteditor.SelectionStart
            curSel = frmTextPad.txteditor.SelectionLength
            frmTextPad.txteditor.Text = _
                    frmTextPad.txteditor.Text.Replace(txtfind.Text.Trim, txtreplace.Text.Trim)
'تبديل الاختيار الاصلي
            frmTextPad.txteditor.SelectionStart = curPos
            frmTextPad.txteditor.SelectionLength = curSel
        End Sub
        Private Sub txtfind_TextChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles txtfind.TextChanged
            If txtfind.Text.Length > 0 Then
                btnfind.Enabled = True
                btnfindnext.Enabled = True
            Else
                btnfind.Enabled = False
                btnfindnext.Enabled = False
            End If
            If txtreplace.Text.Length > 0 Then
                btnreplace.Enabled = True
                btnreplaceall.Enabled = True
            Else
                btnreplace.Enabled = False
                btnreplaceall.Enabled = False
            End If
        End Sub
```

```vb
    Private Sub txtreplace_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles txtreplace.TextChanged
        If txtreplace.Text.Length > 0 Then
            btnreplace.Enabled = True
            btnreplaceall.Enabled = True
        Else
            btnreplace.Enabled = False
            btnreplaceall.Enabled = False
        End If
    End Sub
    Private Sub frmfind_Activated(ByVal sender As Object, ByVal e As System.EventArgs) Handles
Me.Activated
'التاكد من ان الأزرار بشكل أولي غير فعالة
        txtfind_TextChanged(sender, e)
    End Sub
End Class
```

مناقشة الكود

يتفحص هذا الأمر قيمة اداة صندوق الاختيار Checkcasesensitive والتي تحدد فيم اذا سيكون البحث مع التحسس لحالة الأحرف ويستدعي النموذج المناسب لطريقة الفهرس IndexOf, المعامل الأول لهذه الطريقة هو النص الذي نبحث عنه,والمعامل النسي الثاني هو موديل البحث searchmode وقيمته عضو في العداد"مقارنة النصوص" StringComparison: فالنظامي Ordinal من اجل حالة التحسس case-sensitive و OrdinalIgnoreCase حالة تجاهل النظامي من اجل عمليات البحث الغير حساسة لحالة الأحرف.فإذا ما وجدت الطريقة IndexOf النص فان البرنامج يختاره باستدعاء طريقة الأداة "اختيار" مع المعاملات النسبية المناسبة و اذا لم تجد النص فإنها تعرض رسالة.لاحظ انه بعد نجاح عملية الإيجاد ان الأزرار على الفورم الثانوية تصبح فعالة.وكود زر"إيجاد التالي" مشابه تماما ولكن يبدأ البحث عند الحرف الذي يلي الاختيار الحالي,فالطريقة تعمل على إيجاد النسخة الثانية من نفس النص(راجع الكود)

أما الزر "تبديل بـ" فيعمل على استبدال الاختيار الحالي بنص الاستبدال ومن ثم يعمل على إيجاد النسخة الثانية من نص الإيجاد. اما الزر "تبديل الكل" فانه يعمل على تبديل جميع نسخ كلمة البحث في المستند,(راجع الكود)

الطريقة Replace "تبديل" حساسة لحالة الأحرف case-sensitive وهذا يعني انها تستبدل نسخ معامل النسي للبحث في النص الذي لديه نفس التهجئة spelling كما في معامله النسي الأول,من اجل البحث الغير حساس لحالة الأحرف عليك كتابة الكود. بشكل اختياري بإمكانك استخدام وظيفة الاستبدال الجاهزة لإتمام عمليات البحث غير لحساسة لحالة الأحرفcase-insensitive كما هو مبين في الكود التالي:

```vb
Replace(frmTextPad.txteditor.Text, txtfind.Text.Trim, txtreplace.Text.Trim, , , CompareMethod.Text)
```

المعامل النسي الأخير والذي هو اختياري يحدد فيم اذا البحث سيكون case-sensitive حساس (مقارنة ثنائية CompareMethod.Binary) او case-insensitive غير حساس(مقارنة نصية CompareMethod.Text)

الأمر تراجع يتم تنفيذه مع استدعاء الطريقة تراجعUndo ولان الطريقة Undoتعمل كعقدة تبديل أو تمفصل toggle علينا أيضا أن جعل captionعنوانها عقدة تمفصل من تراجع إلى عكس التراجع (والعكس بالعكس) في كل يتم فيها تفعيل هذا الأمر.

إذا حررت النص بعد عملية تراجع فلا تستطيع أن تعكس آخر عملية تراجع وهذا يعني انه حالما يتغير المحتوى لصندوق النص فان عنوان أمر "التراجع وعكس التراجع" يجب أن يصبح"تراجع" حتى ولو كان في الوقت المعني عكس التراجع .الأمر عكس التراجع متاح فقط بعد عملية تراجع وقبل تحرير النص لذا كيف سنعرف انه تم تحرير النص؟ تطلق أداة صندوق النصTextBox الحدث TextChangedفي كل مرة يتغير محتواها.سنستخدم هذا الحدث لإعادة تخزين العنوان captionللأمر "تراجع/عكس التراجعUndo/Redo" إلى تراجع"Undo "(راجع الكود)

أداة صندوق النص لا تستطيع أن تزودك بالكثير من عمليات التراجع,فهي بهذا تختلف عن برنامج الورد الذي يحفظ مسار أفعال المستخدم(مثل الإدخال،الحذف ،عمليات التبديل وهكذا)ومن ثم فانه من الممكن التراجع عنها بخطوات فإذا كنت تريد عمليات تراجع أكثر استخدم أداة RichTextBoxصندوق النص الغني ،باستطاعة هذه الأداة أن تعرض النصوص المنسقة بالإضافة إلى إمكانية استخدامها كأداة صندوق نص متطور,على فكرة العمل على إعداد عنوان احد بنود القائمة من ضمن الحدث TextChanged"تغير النص" فيها إسراف overkill أو مبالَغة لان هذا الحدث يحدث في كل مرة يضغط فيها المستخدم على مفتاح,ولكن العملية لا تستهلك وقت إطلاقا، ولا تُفقِد التطبيق سرعة الاستجابةresponsive . الاختيار الأفضل سيكون الحدث DropDownOpening لأحد بنود القائمة edit item الذي سيتم إطلاقه في كل مرة يفتح فيها المستخدم قائمة التحرير.

التقاط كبس الزر Capturing Keystrokes

لأداة صندوق النص حدث مفرد ومميز وهو الحدث TextChanged"تغير النص" والذي يتم إطلاقه كل مرة يتم تغير نص الأداة إما بسبب عمل المستخدم على كتابة حروف أو عملية اللصق يوجد حدث آخر مشترك في برمجة أداة صندوق النص وهو الحدث KeyPress"كبس مفتاح" والذي يحدث كل مرة يتم فيها ضغط مفتاح ما ويثبت(يبلغ) حرف المفتاح الذي تم ضغطه.تستطيع استخدام هذا الحدث لالتقاط مفاتيح محددة وتعديل سلوك البرنامج بالاعتماد على الحرف الذي تمت كتابته.

على فرض انك تريد استخدام تطبيق المفكرة لتحضير رسائل ليتم إرسالها عبر خط الفاكس telexوكما تعلم الفاكس لا يستطيع إرسال الحروف الصغيرة أو الرموز الخاصة فعلى المحرر أن يحول النص إلى الحالة الكبير للحروف ويستبدل الرموز الخاصة بالنصوص المكافئة لها:DLR من اجل $و ATمن اجل@ و O/O من اجل% و BPTمن اجل # وAND من اجل &.تستطيع أن تعدل السلوك الافتراضي لأداة صندوق النص من ضمن الحدث KeyPressبحيث يعمل على تحويل هذه الحروف كما كتبها المستخدم.

من خلال التقاط كبس المفاتيح,تستطيع أن تعالج البيانات كما تمت عملية إدخالها, في الوقت الحقيقي,مثلا تستطيع التأكد من أن صندوق النص يقبل فقط قيم عددية أو الرموز الست عشريةhexadecimal characters وترفض ما عدا ذلك,لمعالجة محرر ما لتحضير نص من اجل النقل transmission عبر الفاكس,استخدم معالج حدث KeyPress "كبس الزر" كما هو موضح في الكود(راجع الكود)

تختبر الجملة الأولى في معالج حدث"كبس المفتاح" المفتاح الذي تم ضغطه وتخرج إذا كان هذا المفتاح مفتاح تحرير خاص(حذفDelete, تراجع خطوةBackspace , Ctrl+V وهكذا) في هذه الحالة يخرج معالج الحدث دون أن يقوم بأي فعل,الخاصية KeyChar للمعامل النسي لحدث KeyPress"كبس المفتاح" تثبت أو(تعمل على تبليغ)المفتاح الذي تم ضغطه,ويعمل الكود على تحويله إلى نص ومن ثم يستخدم عبارة Caseالحالة لمعالجة المفاتيح المكبوسة بشكل مستقل. إذا كبس المستخدم المفتاح $ مثلا فان الكود سيعرض DLR و إذا لم يتم كبس رمز خاص فان الكود يعرض الحروف المكبوسة كما هي من خلال شرط الحالة Elseلعبارة الاختيار

ملاحظة: إلغاء كبس المفاتيح Cancelling Keystrokes

قبل أن تخرج من معالج الحدث،يتوجب عليك أن تقتل"تنهي’’kill‘‘" المفتاح الأصلي الذي تم كبسه,لذا فانه لن يتم السماح له بالظهور على الأداة.تستطيع عمل هذا بواسطة إعداد الخاصية Handled"تمت المعالجة" إلى True صح والتي تجبر فيجوال بيسك أن ليس عليه معالجة "كبس المفتاح" بعد الآن.فإذا حذفت هذه العبارة فان الحرف الخاص سيتم كتابته مرتين:مرة بتنسيق تحويلهم (DLR$ ، AT@,وهكذا)ومرة أخرى كأحرف نظامية.تستطيع أيضا أن تعمل على إعداد الخاصية SuppressKeyPressإلى صح لإلغاء كبس مفتاح ما،وبالتالي فان ما سيحدث هو ان Common Language Runtime (CLR)لغة التنفيذ المشتركة لن تمرر كبس المفتاح إلى الأداة المناسبة.

التقاط مفاتيح الوظائف Capturing Function Keys

توجد ميزة مشتركة أخرى في تطبيق "قارئ النصوص" وهي إسناد عمليات خاصة إلى مفاتيح الوظائف.تطبيق المفكرة في الويندوز مثلا يستخدم مفتاح الوظيفة F5لإدخال التاريخ الحالي عند موضع المشيرة. تستطيع أن تعمل نفس الشيء مع تطبيق المفكرة هنا ولكن لا تستطيع استخدام حدث KeyPressكبس المفتاح حيث أن المعامل النسبي KeyCharلا يثبت مفاتيح الوظائف. الأحداث التي تستطيع أن تلتقط مفاتيح الوظائف هي الأحداث KeyUp وKeyDown.ما تختلف به هذه الأحداث عن حدث كبس المفتاح هو أن هذان الحدثان لا يثبتان الحرف الذي تم ضغطه,فبدلا من ذلك يثبتان كود المفتاح(العدد الخاص الذي يميز كل مفتاح على لوحة المفاتيح ويعرف أيضا بكود ***scancode*** التفحص) من خلال الخاصية e.KeyCode.

كود المفتاح keycodeيميز uniqueبالنسبة لكل مفتاح وليس لكل حرف,فالحروف في الحالة الكبيرة والحالة الصغيرة لكل منها قيم أسكي ASCIIمختلفة ولكن لها نفس كود المفتاح لأنها على نفس المفتاح,مثلا العدد 4 والرمز $ لهما نفس كود المفتاح لأن نفس المفتاح في لوحة المفاتيح ينتج هذين الحرفين.عندما يتم تثبيت كود المفتاح فان الحدثين KeyDown وKeyUp يثبتان أيضا حالة مفتاح الرفع Shift أو Ctrl و Altمن خلال الخاصيات e.Shift و e.Controlوe.Alt . معالج الحدث KeyUpالمبين في الكود يستخدم مفاتيح الوظائف F5 وF6 لإدخال التاريخ الحالي والوقت في المستند.ويستخدم أيضا F7 وF8 لإدخال نصوص معرفة مسبقا في المستند.

يستخدم الويندوز العديد من مفاتيح الوظائف (مثلا المفتاح F1 للتعليمات أو المساعدة)ولن تعمل على تعديل وظائفها الأصلية.مع القليل من الجهد تستطيع أن تعمل على تزويد المستخدم بصندوق حوار والذي يدع المستخدم من إسناد النصوص الخاصة به إلى مفاتيح الوظائف,من المحتمل أن تأخذ بعين الاعتبار حالة الرفع Shift أو Ctrl التحكم أو Altالتحويل على التسلسل لتستكشف فيما إذا سيتم ضغط مفاتيح التعديل هذه modifier keys مع مفتاح آخر.استخدم المعامل AND مع الخاصية المناسبة للمعامل النسبي e.

خاصية الإكمال التلقائي Auto-complete Properties

احد إعدادات خاصية الإدخال لأداة صندوق النص هي خاصيات الإكمال التلقائي(الآلي) هل لاحظت كيف يعمل مستكشف الانترنت على إرشادك حالما تبدأ بكتابة عنوان ما في صندوق نص اسم المستخدم(أو في شريط عنوان المستعرض) تستطيع بسهولة أن تعالج مثل هذه الصناديق باستخدام أداة صندوق نص ذات سطر واحد وخواص الإكمال التلقائي.بشكل أساسي عليك أن تخبر أداة صندوق النص كيف عليها الطلب من المستخدم من خلال نصوص مطابقة الأحرف المكتوبة مسبقا على الأداة ومن أين ستأتي عملية المطابقة ومن ثم فان باستطاعة الأداة عرض قائمة منسدلة بالنصوص التي تبدأ بالحروف التي تمت كتابتها حاليا بواسطة المستخدم،وبإمكان المستخدم إما الاستمرار في الكتابة(في هذه الحالة قائمة الخيارات تصبح اقصر) أو يختار بند من القائمة,يتم تطبيق خاصيات الإكمال التلقائي بالنسبة لأدوات صندوق النص ذات السطر المفرد فقط فهي لا تؤثر في أدوات صناديق النصوص المتعددة الاسطر.

في عدة حالات ,الإكمال التلقائي لأداة صندوق النص أكثر فعالية من أداة ComboBoxالصندوق المركب وسوف تفضل أداة صندوق النص وسترى أن أداة الصندوق المركب تدعم أيضا خاصيات الإكمال التلقائي,لأنها تجعل الأداة سهلة الاستعمال وباستخدام لوحة المفاتيح فقط.تحدد الخاصية AutoCompleteMode "صيغة الإكمال التلقائي" الماهية والكيفية التي سيتم من خلالها سؤال المستخدم, و إعداداتها عضو في عداد "صيغة الإكمال التلقائي AutoCompleteMode enumeration "(Autosuggest الاقتراح التلقائي, AutoAppend الإلحاق الآلي, AutoAppend,الإلحاق والاقتراح الآلي AutoSuggestAppend, ولا شيءNone) في صيغة AutoAppend mode الإلحاق تختار أداة صندوق النص بند المطابقة الأول في قائمة الاقتراحات وتكمل النص. في صيغة AutoSuggestAppend mode "الإلحاق والاقتراح الآلي" تقترح الأداة بند المطابقة الأول في القائمة كما سبق ولكنها توسع القائمة في صيغة الاقتراح الآليAutoSuggest mode , تفتح الأداة بكل بساطة القائمة مع البنود الموافقة ولكن لا تختار أي بند. لأدوات صناديق النص النظامية يتم وضع خاصية صيغة الإكمال التلقائي AutoCompleteModeإلى خطأ(False) وتحدد خاصية مصدر الإكمال التلقائي AutoCompleteSource من أين تأتي قائمة الاقتراحات وقيمتها عضو في عداد مصدر الإكمال التلقائي AutoCompleteSourceالمبين في الجدول التالي:

البند	الشرح
مصادر جميع الأنظمةAllSystemSources	البنود المقترحة هي أسماء مصادر النظام
عناوين ويب(انترنتAllUrl)	البنود المقترحة هي العناوين URLsالتي تم زيارتها بواسطة الكمبيوتر الهدف ,ولا تعمل إذا حذفت الصفحات التي تم عرضها حديثا
مصدر مخصص CustomSource	البنود المقترحة تأتي من جمع مخصصcustom collection
ملف نظام FileSystem	البنود المقترحة هي أسماء ملفات
قائمة التاريخ(ملفات المؤقتةHistoryList)	تأتي بنود الاقتراح من قائمة history تاريخ الكمبيوتر
قائمة الاستخدام الأحدث RecentlyUsedList	البنود المقترحة تأتي من مجلد تم استخدامه حديثا
لا يوجد None	الأداة لا تقترح أي بند

لتوضيح خصائص الإكمال التلقائي سنقوم بعمل تدريب,وهذا التدريب يمنحك إمكانية إعداد صيغة الإكمال التلقائي والمصدر من اجل أداة صندوق نص وحيد السطر فصندوق النص الأعلى يستخدم قائمة كلمات مخصصة بينما الصندوق الأدنى يستخدم مصادر معد مسبقا(جاهزة) (ملفات نظامfile system, عناوين انترنت URLs, وهكذا)

تدريب

صمم واجهة المستخدم لتصبح مشابهة للشكل المعروض ضع عليه الأدوات المبينة في الشكل ولا تنسى كما ذكرنا سابقا أن تجعل خصائص الفورم من اليمين إلى اليسار صحيحة من اجل الواجهة العربية بالنسبة لأزرار التبديل اجعل خاصية checked=true بالنسبة "العناوين الانترنت"و"اقتراح "وكما هو مبين في الشكل

الاتمام الالي لصناديق النصوص

ادخل اسم اللغة

صندوقي نص

اختار عنوان انترنت او ملف

صيغة الاكمال التلقائي

مصدر الاكمال التلقائي

اقتراح

بدون

قائمة مؤقتة

ملف نظام

اقتراح والحاق

الحاق

قائمة حديثة

عناوين انترنت

ثلاث ادوات عنوان

اعدادات صيغة الاكمال التلقائي يتم تطبيقها على صندوقي النص بينما اعدادات مصدر الاكمال التلقائي يتم تطبيقها على صندوق النص الذي بالاسفل

GroupBox صندوق جمع عدد٢

radiobutton ازرار تبديل

إليك الكود كاملا لاحظ أن الاسم البرمجي لكل أداة يعبر عن نفسه إذا أردت غير الأسماء البرمجية لأدواتك كما هي مبينة في الكود التالي

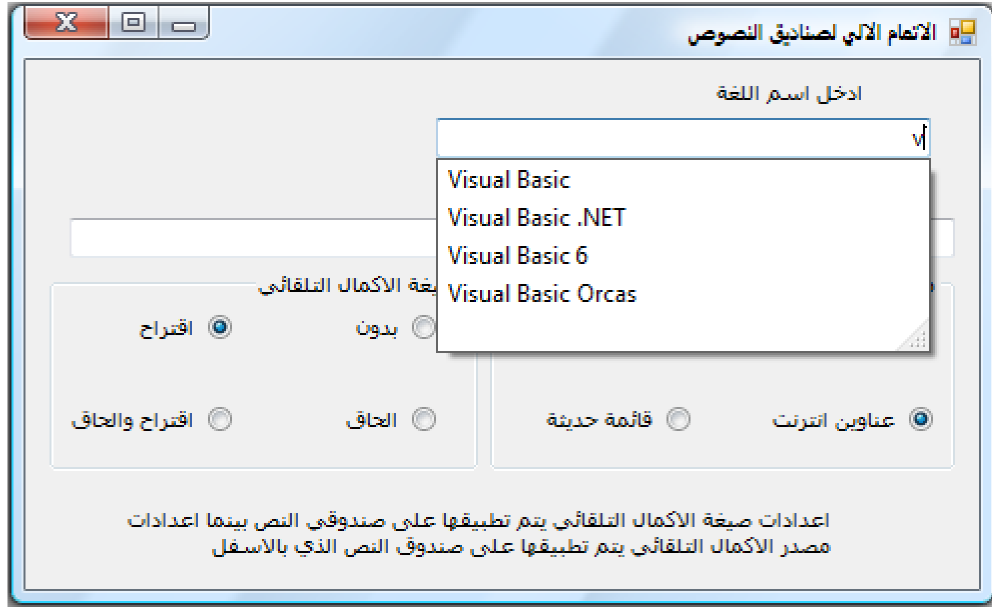
```
Public Class autocomplete
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
'إعداد خاصيات الإكمال التلقائي من اجل صندوق النص العلوي(في حدث تحميل الفورم)
        Dim knownWords As New AutoCompleteStringCollection
        knownWords.Add("Visual Basic Orcas")
        knownWords.Add("Visual Basic .NET")
        knownWords.Add("Visual Basic 6")
        knownWords.Add("Visual Basic")
        knownWords.Add("Framework")
        TextBox1.AutoCompleteCustomSource = knownWords
        TextBox1.AutoCompleteSource = AutoCompleteSource.CustomSource
        TextBox1.AutoCompleteMode = AutoCompleteMode.Suggest
'إعداد خاصيات الإكمال التلقائي بالنسبة لصندوق النص الذي في الاسفل
        TextBox2.AutoCompleteSource = AutoCompleteSource.RecentlyUsedList
        TextBox2.AutoCompleteMode = AutoCompleteMode.Suggest
    End Sub
    Private Sub rbNone_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
'اجراء معالجة تغير اختبار ازرار التبديل بالنسبة لأزرار التبديل التابعة لصيغة الإكمال التلقائي
        If rbNone.Checked Then
            TextBox1.AutoCompleteMode = AutoCompleteMode.None
            TextBox2.AutoCompleteMode = AutoCompleteMode.None
        End If
        If rbAppend.Checked Then
            TextBox1.AutoCompleteMode = AutoCompleteMode.Append
            TextBox2.AutoCompleteMode = AutoCompleteMode.Append
        End If
        If rbSuggest.Checked Then
            TextBox1.AutoCompleteMode = AutoCompleteMode.Suggest
            TextBox2.AutoCompleteMode = AutoCompleteMode.Suggest
        End If
        If rbSuggestAppend.Checked Then
            TextBox1.AutoCompleteMode = AutoCompleteMode.SuggestAppend
            TextBox2.AutoCompleteMode = AutoCompleteMode.SuggestAppend
        End If

    End Sub

    Private Sub rbFileSystem_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles rbFileSystem.CheckedChanged, rbAllURL.CheckedChanged,
rbHistoryList.CheckedChanged, rbRecentList.CheckedChanged
'حدث معالجة ازرار التبديل بالنسبة لأزرار التبديل التابعة لمصدر الإكمال التلقائي
        If rbAllURL.Checked Then TextBox2.AutoCompleteSource = AutoCompleteSource.AllUrl
        If rbFileSystem.Checked Then TextBox2.AutoCompleteSource = AutoCompleteSource.FileSystem
        If rbHistoryList.Checked Then TextBox2.AutoCompleteSource =
AutoCompleteSource.HistoryList
```

```
        If rbRecentList.Checked Then TextBox2.AutoCompleteSource =
AutoCompleteSource.RecentlyUsedList
    End Sub
End Class
```

بعد أن تكمل تصميم الواجهة وبعد أن تدخل الكود سيظهر لك تنفيذ البرنامج الخيارات التالية في صندوق النص العلوي بعد كتابة الحرف v

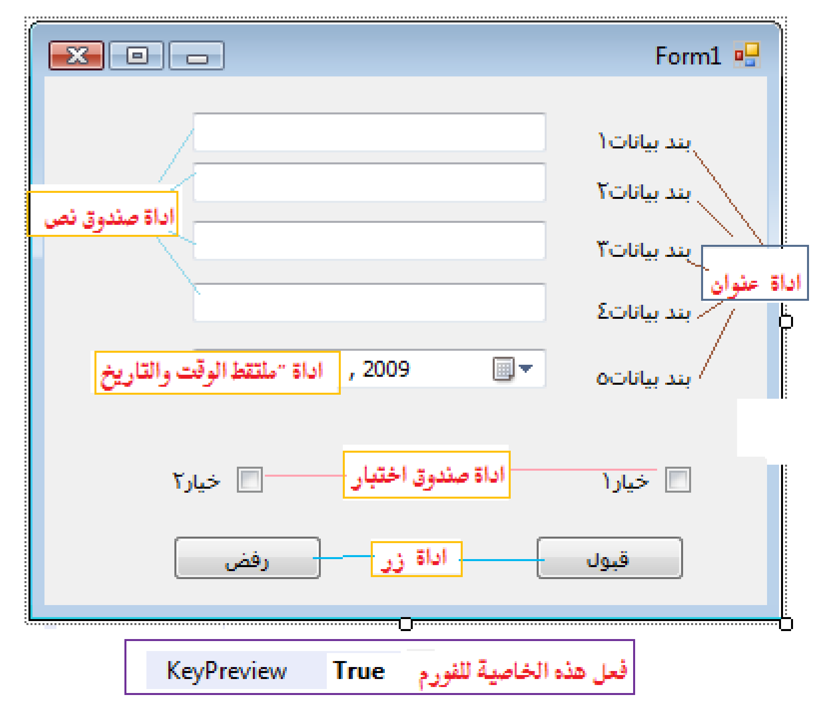


وإذا كتبت h في الصندوق السفلي سيظهر لك ما هو مبين في الشكل التالي

إذا وضعت " AutoCompleteSource مصدر الإكمال التلقائي" CustomSource " مصدر مخصص" عليك أن تعمل على ملئ الكائن AutoCompleteStringCollection بالاقتراحات المرغوبة وتعمد على إسناده إلى الخاصية AutoCompleteCustomSource . الجمع هو مجرد مجموعة من النصوص كما يظهر حدث تحميل الفورم حيث عملنا على إعداد هذا الجمع وملأناه بلغات البرمجة واستخدمناه مع صندوق النص الأول.

ملاحظة: Data-Entry Applications

تحوي التطبيقات العملية النموذجية على عدد هائل من الفورمات لإدخال البيانات والعنصر الأكثر أهمية من اجل إدخال البيانات هو أداة صندوق النص,وعمليات الإدخال للبيانات فعالة جدا باستخدام لوحة المفاتيح ويجب التمكن من استخدام تطبيقاتك بدون اللجوء إلى الفارة . عمليات إدخال البيانات المختلطة لا يكتب لها الاستمرارية بدون المفتاح Enter وتستطيع الوصول إلى هذا المفتاح عند نهاية كل عملية,فالواجهة الفعالة يجب أن تعمل على إضافة بعض الذكاء إلى "keystroke كبس المفتاح":أي أن المفتاح Enterيجب أن يتم انجازه بوضوح أو بطريقة مماثلة لوظائفه التي يعرفها كلنا.عندما تعمل على إدخال بيانات مثلا يجب أن تأخذ المستخدم إلى الأداة التالية وحسب ترتيب التاب Tab order(مفتاح التاب).اعتبر نافذة إدخال بيانات مثل في الشكل التالي,والتي تحتوي على العديد من أدوات صندوق النص,و أداة DataTimePicker" التقاط الوقت والتاريخ DataTimePicker" من اجل إدخال التواريخ, وأداتي CheckBoxصندوق اختيار هذه هي الفورم الرئيسية لإدخال بيانات بسيطة (صمم الفورم كما هو مبين في الشكل التالي وهي عملية بسيطة جدا)



يستخدم التطبيق مفتاح Enter:كل مرة يتم الضغط على مفتاح الإدخال Enter ينتقل التركيز focusإلى الأداة التالية في ترتيب مفتاح التاب Tabحتى ولو كانت الأداة الحالية هي أداة صندوق اختيارCheckBox ،فعملية keystroke " كبس المفتاح" لا تغير حالة أدوات صندوق الاختيار بل ببساطة تنقل التركيز إلى الأمام. تستطيع برمجة الحدث KeyUpلكل أداة من اجل التفاعل مع مفتاح الإدخالEnter, ولكن هذه المقاربة approach يمكن أن تقود إلى مشاكل في الصيانة إذا كنت خطط لإضافة أدوات جديدة إلى فورم موجود.فالمقاربة الأفضل هي اعتراض interceptسبيل keystrokeكبس مفتاح Enterعلى مستوى الفورم قبل أن يصل إلى الأداة,. لفعل هذا يتوجب عليك وضع الخاصية KeyPreview"عرض المفاتيح مسبقا" للفورم إلى صح وهذا الإعداد يسبب في إطلاق أحداث المفاتيح على مستوى الفورم أولا ومن ثم للأداة التي يكون التركيز عليها. في الحقيقة تمنحك هذه الخاصية إمكانية معالجة " keystrokes كبس مفاتيح"للعديد من الأدوات في نفس الوقت. عالج الحدث KeyUpلتطبيق إدخال البيانات للفورم الرئيسية وعلى مستوى هذه الفورم يعترض كبس زر الإدخال ويتفاعل معه بنقل التركيز إلى الأداة التالية في ترتيب التاب بدعم الطريقة ProcessTabKey.هذه الطريقة simulatesتحاكي معالجة مفتاح التنقل"Tab key التاب" ويتم استدعاءها من خلال معامل نسبي وحيد,والذي هو قيمة Booleanمنطقية:صح :نقل التركيز إلى الأمام وخطأ:نقل التركيز إلى الخلف:إليك كود المشروع كاملا:

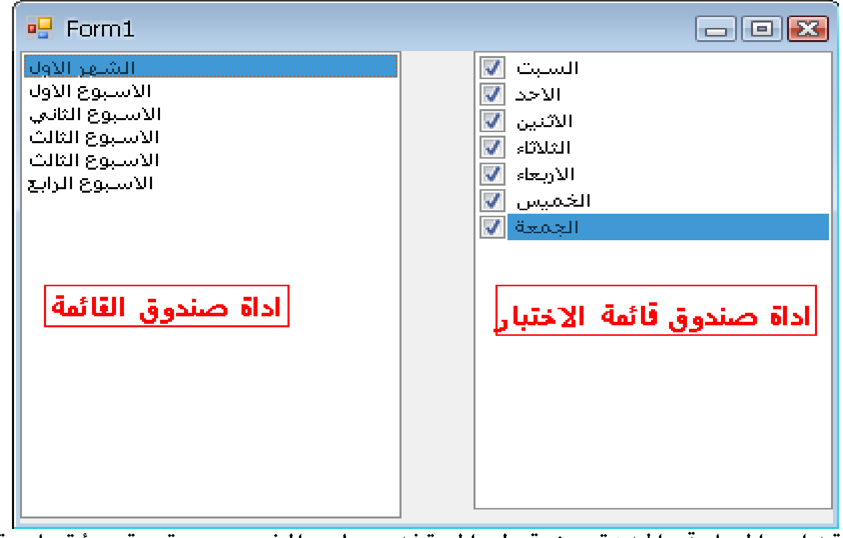
```
Public Class Form1
    Private Sub Form1_KeyDown(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyEventArgs) Handles Me.KeyDown
        'إذا ضغط المستخدم المفتاح انتر'
        If e.KeyCode = Keys.Enter And Not (e.Alt Or e.Control) Then
            e.SuppressKeyPress = True
'تأكد من أن الأداة الفعالة هي صندوق نص'
'لا تستخدم المفتاح انتر كمفتاح تنقل عندما يكون التركيز على الزر'
            If Me.ActiveControl.GetType Is GetType(TextBox) Or _
                    Me.ActiveControl.GetType Is GetType(CheckBox) Or _
                    Me.ActiveControl.GetType Is GetType(DateTimePicker) Then
'استخدم مفتاح الرفع+انتر للتنقل للخلف بترتيب التاب'
                If e.Shift Then
                    Me.ProcessTabKey(False)
                Else
                    Me.ProcessTabKey(True)
                End If
            End If
        End If
    End Sub
    Private Sub button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
        Dim msg As String = ""
        msg = "البيانات بند 1: " & TextBox1.Text.Trim & vbCrLf & _
                    "البيانات بند 2: " & TextBox2.Text & vbCrLf & _
                    "البيانات بند 3: " & TextBox3.Text & vbCrLf & _
                    "البيانات بند 4: " & TextBox4.Text & vbCrLf
        msg &= "التاريخ المختار: " & DateTimePicker1.Value.ToShortDateString & vbCrLf
        msg &= "خيار 1: " & CheckBox1.Checked.ToString & vbCrLf
        msg &= "خيار2: " & CheckBox2.Checked.ToString & vbCrLf
        MsgBox(msg)
        ClearFields()
        TextBox1.Focus()
    End Sub
    Private Sub ClearFields()
        TextBox1.Clear()
```

```
        TextBox2.Clear()
        TextBox3.Clear()
        TextBox4.Clear()
        DateTimePicker1.Value = Now.ToShortDateString
        CheckBox1.Checked = False
        CheckBox2.Checked = False
    End Sub
    Private Sub button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button2.Click
        ClearFields()
        TextBox1.Focus()
    End Sub
End Class
```

يوجد شيئان يمكن أن تكون قد لاحظتهما فيما يخص معالج الحدث: الأول انه لا يستجيب لمفتاح Enterالإدخال إذا تم ضغطه مع مفتاح التحويل "Alt" أو مفتاح التحكم "Ctrl" بينما مفتاح الرفع Shiftمن الناحية الأخرى هو المفتاح المستخدم للتحكم بالاتجاه في ترتيب التنقلTab. ينتقل التركيز إلى الأمام باستخدام كبس مفتاح الإدخال وينتقل التركيز إلى الخلف باستخدام مفتاح الإدخال مع مفتاح الرفع Shift + Enterسويا.وتمت معالجة التركيز بالنسبة لجميع الأدوات على الفورم ما عدا الأزرار حيث انه عندما يكون التركيز على احد الأزرار فان ضغط مفتاح الإدخال يستجيب الزر كما هي العادة.

أدوات الصندوق المركب ComboBoxوصندوق قائمة الاختيار CheckedListBox وصندوق القائمةListBox

تجلب هذه الادوات قوائم من الاختيارات والتي يستطيع المستخدم ان يختار واحد منها او أكثر.يوضح الشكل التالي كلا من صندوق القائمة وصندوق قائمة الاختيار



يشغل صندوق القائمة مقدار المساحة المحددة من قبل المستخدم على الفورم ويتم تعبئتها بقائمة من البنود,اذا كانت قائمة البنود أطول من ان تناسب إظهارها على الأداة يظهر شريط انزلاق شاقولي بشكل آلي.

اداة صندوق نص الاختيار هي توسع وتنوع لأداة صندوق القائمة وهي مماثلة لأداة صندوق النص ولكن صندوق الاختيار يظهر أمام كل بند ويستطيع المستخدم ان يختار أي عدد من البنود بوضع علامة الصح في صندوق الاختيار الذي أمام البند وكما تعلم فانك تستطيع ان تختار العديد من البنود من اداة القائمة بواسطة ضغط مفتاح الرفع Shiftومفتاح التحكمCtrl.

اداة صندوق المركب تحتوي ايضا على العديد من البنود ولكن بشكل عام تشغل مساحة اقل على الشاشة وهي اداة صندوق قائمة قابل للتوسعة(التمدد)حيث يستطيع المستخدم ان يوسعها من اجل عملية الاختيار ومن ثم فانها تنكمش بعد ان تتم عملية الاختيار. الفائدة الحقيقية من اداة الصندوق المركب هي السماح للمستخدم بإدخال معلومات جديدة في الصندوق المركب بدلا من إجباره على الاختيار من بنود مجدولة.

لإضافة بنود وقت التصميم,حدد خاصية Itemsالبنود في نافذة الخصائص للأدوات Properties window ومن ثم اضغط على زر التبديل ستظهر لك نافذة وهي نافذة محرر جمع النصوص String Collection Editor window والذي تستطيع به ان تضيف البنود التي تريد ان تعرضها في القائمة وكل بند يجب ان يظهر في سطر منفصل واسطر النصوص الفارغة سينتج عنها اسطر فارغة في القائمة وهذه البنود ستظهر في القائمة عندما يتم تحميل الفورم (ولكن تستطيع إضافة المزيد او تحذف من الموجود) من خلال الكود في أي وقت,تظهر البنود بنفس ترتيب إدخالها في نافذة محرر جمع النصوص ما لم يتم تفعيل خاصية الترتيب Sortedللأداة في هذه الحالة يتم ترتيب البنود بشكل أوتوماتيكي بغض النظر عن الترتيب الأولي الذي حددته لهم.

الخصائص الأساسية Basic Properties

الارتفاع لصحيح IntegralHeight

هذه الخاصية هي قيمة منطقية (صح/خطأ)والتي تشر الى ان ارتفاع الأداة سيتم تعديله أم لا،لتجنب العرض الجزئي للبند الأخر فعندما يتم وضعها الى صح فان الارتفاع الفعلي للأداة يتغير بمضاعفة الارتفاع بمقدار ارتفاع السطر المفرد.ولذا فانه سيتم عرض عدد صحيح من الصفوف في كل الأوقات.

البنود Items

خاصية البنود هي مجمع collectionوالذي يحفظ بنود الأداة . تستطيع وقت التصميم ان تملأ هذه القائمة من خلال نافذة محرر مجمع النصوص.اما وقت التنفيذ تستطيع الوصول ومعالجة البنود من خلال الطرق والخصائص لمجمع البنود والذي تم شرحه باختصار.

تعدد الأعمدة MultiColumn

تستطيع اداة صندوق النص ListBoxان تعرض بنودها في عدة أعمدة اذا عملت على وضع خاصية "تعدد الأعمدة Multicolumn"الى الفعل True . المشكلة في صناديق القوائم المتعددة الأعمدة هي انك لا تستطيع ان تحدد العمود الذي سيظهر فيه كل بند.صناديق القوائم ListBoxesذات البنود العديدة وخاصية Multicolumn"تعدد الأعمدة"المفعّلة True، تتوسع بشكل أفقي وليس بشكل عمودي.فشريط الانزلاق الأفقي سيتم إلحاقه بصندوق القائمة المتعددة الأعمدة, لذا فان المستخدم يستطيع ان يجلب أي عمود الى العرض,وهذه الخاصية غير مطبقة بالنسبة لأداة الصندوق المركبComboBox .

صيغة الاختيار SelectionMode

هذه الخاصية التي يتم تطبيقها على أدوات صندوق القائمة ListBoxوصندوق قائمة الاختيار CheckedListBox فقط تحدد كيف يستطيع المستخدم أن يختار بنود قائمة .القيم المتاحة لمكونات هذه الخاصية من عداد صيغة الاختيار SelectionMode مبينة في القائمة التالية:

القيمة	الوصف
بدون None	لا يتم السماح لأي اختيار على الإطلاق
واحد One	(الافتراضي) يمكن اختيار بند وحيد فقط
متعددة بسيطة MultiSimple	اختيار متعدد بسيط: (نقر الفارة) أو ضغط المسطرة spacebar لاختيار أو عدم اختيار بند من القائمة .عليك ضغط جميع البنود التي تريد اختيارها.
متعدد موسع: MultiExtended	اختيار متعدد موسع: اضغط مفتاح Shiftالرفع وانقر بالفارة (او اضغط واحد من مفاتيح الأسهم)لتوسيع الاختيار.هذه المعالجة تبرز جميع البنود بين البند المختار مسبقا وبند الاختيار الحالي.اضغط مفتاح التحكم Ctrlوانقر بالفارة لاختيار أو عدم اختيار بنود مفردة ف القائمة.

الترتيب *Sorted*

عندما تكون هذه الخاصية صح تبقى البنود مرتبة كل الأوقات.القيمة الافتراضية هي "غير فعال"لأنها تأخذ وقت أطول لإدخال لبنود الجديدة في مكانها المناسب.يمكن ان يتم إعداد هذه الخاصية وقت التصميم ووقت التنفيذ ايضا. يتم ترتيب البنود في أداة صندوق القائمة بشكل تصاعدي وبحالة الحساسية للأحرف فالأحرف الكبيرة تظهر قبل مكافئتها من الأحرف الصغيرة ولكن كلا الحروف في الحالة الكبيرة والصغيرة تظهر مع بعضها بشكل متتابع. صندوق القائمة لا يمكنه ترتيب البيانات العددية فالعدد 10 سيظهر قبل العدد5 لان النص 10 اصغر من النص 5 أما إذا تم تنسيق الأرقام كـ010 و005 فسيتم ترتيبها بالشكل الصحيح.

النص Text

تعود خاصية النص بالنص المختار على الأداة,على الرغم من إمكانية إعداد خاصية النص للصندوق المركب وقت التصميم ولكن هذه الخاصية متاحة فقط وقت التنفيذ بالنسبة للأدوات الأخرى,لاحظ ان البنود لا تحتاج لان تكون نصوص.بشكل افتراضي كل بند هو كائن ومن اجل كل كائن فان الأداة تعرض نصوص,والذي هو نفس النص المعاد بواسطة الطريقة ToString"الى نص"

معالجة مجمع البنود Manipulating the *Items* Collection

لمعالجة أداة صندوق النص من خلال تطبيقك عليك ان تكون قادر على عمل التالي:

1- إضافة بنود إلى القائمة
2- إزالة بنود من القائمة
3- إمكانية الوصول إلى البنود المستقلة في القائمة

يتم تمثيل البنود في القائمة بواسطة مجمع البنود**Items collection** .وانك تستخدم أعضاء مجمع البنود من اجل إمكانية الوصول إلى بنود الأداة ومن اجل إضافة أو إزالة بنود .تعرض خاصية البنود الأعضاء القياسية للمجمع,والتي سيتم شرحها فيما بعد في هذا المقطع. فكل عضو ف مجمع البنود هو كائن .وفي معظم الحالات نستخدم أدوات صندوق القائمة لتخزين نصوص ولكن من الممكن تخزين كائنات .عندما تضيف كائن الى اداة صندوق القائمة فانه سيتم عرض نص في السطر الموافق على الأداة و هو النص المعاد بواسطة طريقة الكائن **ToString**"الى نص".و هي خاصية الكائن التي سيتم عرضها بشكل افتراضي.

تستطيع ان تعرض أي خاصية أخرى للكائن وذلك بوضع خاصية الأداة ValueMemberالى اسم الخاصية.فإذا أضفت كائن خط Font وكائن مربع Rectangleالى مجمع البنود بالعبارات التالية:

```
ListBox1.Items.Add(New Font("Verdana", 12, FontStyle.Bold))
        ListBox1.Items.Add(New Rectangle(0, 0, 100, 100))
```

عندها سيظهر النص التالي في أول سطرين من الأداة:

[Font: Name=Verdana, Size=12, Units=3, GdiCharSet=1, gdiVerticalFont=False]
{X=0, Y=0, Width=100, Height=100}

مهما يكن تستطيع الوصول الى أعضاء كلا الكائنين لان صندوق القائمة يخزن الكائنين وليس شرحهما (الوصف).العبارة التالية تطبع اتساع كائن المربع (المخرجات الناتجة عن العبارة هي 100)

```
Debug.WriteLine(ListBox1.Items.Item(1).Width)
```

التعبير في العبارة السابقة " ربط متأخر*late-bound*" والذي يعني ان المترجم لا يعرف فيم اذا كان الكائن الأول في مجمع البنود كائن مربع و لا يستطيع التحقق verifyمن اتساع المكون. اذا حاولت استدعاء الخاصية Widthللبند الأول في المجمع ستحصل على استثناء exceptionوقت التنفيذ: ان الكود يحاول الوصول الى عضو مفقود (غير موجود) .والعضو المفقود هو خاصية "الاتساع" Widthلكائن الخط.

الطريقة المناسبة لقراءة الكائنات المخزنة في اداة صندوق القائمة هي تفحص النوع للكائن اولا ومن ثم محاولة الوصول لاستخلاص خاصية ما (او استدعاء طريقة)للكائن ,وفقط اذا كان نوعه مناسب. إليك كيف ستقرأ خاصية الاتساع لكائن المربع:

```
If ListBox1.Items.Item(1).GetType Is GetType(Rectangle) Then
            Debug.WriteLine(CType(ListBox1.Items.Item(1), Rectangle).Width)
        End If
```

الطريقة "إضافة" The *Add* Method

لإضافة بند إلى القائمة استخدم الطريقة "إضافة بند **Items.Add** " او "إدخال بند **Items.Insert** ." الشكل العام لطريقة"إضافة بند" كمايلي:

```
ListBox1.Items.Add(item)
```

عامل parameterالبند هو الكائن الذي تمت إضافته الى القائمة.تستطيع أن تضيف أي كائن إلى أداة صندوق القائمة,ولكن تكون البنود عادة نصوص,الطريقة Add"إضافة" تعمل على إلحاق appendsبند جديد إلى نهاية القائمة مالم تكن خاصية"ترتيب Sorted " تم إعدادها إلى صح True . الحلقة التالية تعمل على إضافة عناصر من مصفوفة الكلمات wordsإلى اداة صندوق القائمة ListBoxككلمة في كل مرة.

```
Dim words(100) As String
  {عبارات مناسبة لتعبئة المصفوفة}
        Dim i As Integer
        For i = 0 To 99
            ListBox1.Items.Add(words(i))
        Next
```

بشكل مشابه تستطيع الدوران على جميع البنود الموجودة على الأداة باستخدام حلقة كالتالية:

```
Dim i As Integer
        For i = 0 To ListBox1.Items.Count - 1
{ statements to process item ListBox1.Items(i) }
```

```
        Next
    End Sub
```

You can also use the For Each . . . Next statement to iterate through the Items collection, as shown here:

تستطيع أيضا استخدام العبارة For Each...Next للدوران على جمع البنود كما هو مبين هنا:

```
Dim itm As Object
        For Each itm In ListBox1.Items
{ process the current item, represented by the itm variable }
        Next
```

عندما تملأ أداة صندوق القائمة بعدد ضخم من البنود,استدعي الطريقة"ابدأ التحديث BeginUpdate " قبل البدء بالحلقة و استدعي الطريقة"أنهي التحديث EndUpdate " عندما تنتهي.هاتان الطريقتان تعملان على تعطيل التحديث المرئي للأداة بينما تملأ الأداة وتسرعان عملية المعالجة بشكل ملحوظ.عندما يتم استدعاء الطريقة "أنهي التحديث EndUpdate " فإنه يتم إعادة تشكيل الأداة مع جميع البنود.

الطريقة "إدخال" The Insert Method

لإدخال بند في موقع محدد استخدم الطريقة "إدخال " Insert " والتي لها الشكل العام التالي:

```
ListBox1.Items.Insert(index, item)
```

عامل parameterالبند item هو الكائن الذي سيتم إضافته،والفهرس index هو موضع البند الجديد,وفهرس البند الأول هو الصفر.لاحظ انك لا تحتاج إلى إدخال البنود عند موقع معين عندما تكون القائمة مرتبة.وإذا عملت ذلك,فإن البنود سيتم إدخالها عند مواضع محددة ولكن القائمة لن تكون مرتبة بشكل تلقائي بعد الآن.

الطريقة "تنظيف" The Clear Method

تعمل هذه الطريقة على إزالة جميع البنود من الأداة. والشكل العام لها:

```
List1.Items.Clear
```

الخاصية "إحصاء" The *Count* Property

هي عدد البنود في القائمة. إذا أردت الوصول إلى البنود باستخدام الحلقة For . . . Next حلقة الإحصاء يجب ان تذهب من 0 الى ListBox.Items.Count - 1 كما هو مبين في مثال طريقة Add "إضافة"

الطريقة"نسخ إلى" The CopyTo Method

تستخلص هذه الطريقة جميع البنود من أداة صندوق القائمة وتخزنها في مصفوفة ممررة إلى الطريقة كمعامل نسبي.
الشكل العام للطريقة "نسخ إلى" هو :

```
ListBox.CopyTo(destination, index)
```

حيث ان الوجهة destinationهي اسم المصفوفة التي ستقبل البنود،والفهرس index هو فهرس عنصر من المصفوفة حيث سيتم تخزين البند الأول.المصفوفة التي تثبت بنود الأداة يجب ان يتم التصريح عنها بشكل صريح ويجب ان تكون كبيرة بما يكفي لحفظ جميع البنود.

الطريقة "إزالة" و"إزالة عند" The *Remove* and *RemoveAt* Methods

لإزالة بند من قائمة,تستطيع بكل بساطة استدعاء طريقة جمع البنود " Remove إزالة " وتمرر الكائن الذي ستتم إزالته كمعامل نسبي. اذا كانت الأداة تحوي نصوص .مرر النص الذي ستتم إزالته . إذا كان نفس النص يظهر في العديد من البنود على الأداة فان النسخة الأولى فقط سيتم إزالتها.
تستطيع أيضا إزالة بند بواسطة تحديد موقعه في القائمة بالطريقة"إزالة عند RemoveAt "والتي تقبل كمعامل نسبي موضع البند الذي ستتم إزالته:

```
ListBox1.Items.RemoveAt(index)
```

العامل"فهرس index " هو ترتيب البند الذي ستتم إزالته والبند الأول ترتيبه هو الصفر 0

الطريقة"المحتويات The Contains Method "

يجب أن لا تخلط بين هذه الطريقة Containsالتابعة لجمع البنود وطريقة Contains التابعة للأداة,فطريقة جمع البنود تقبل كمعامل نسبي كائن ما وتعود بقيمة صح/خطأ والتي تشير فيما إذا كان الجمع يحوي على هذا الكائن.استخدم طريقة Contains لتجنب الإدخالات لكائنات متماثلة في صندوق القائمة.العبارات التالية تضيف نص الى جمع البنود فقط إذا كان النص ليس جزء مسبق من جمع البنود:

```
Dim itm As String = "Remote Computing"
        If Not ListBox1.Items.Contains(itm) Then
            ListBox1.Items.Add(itm)
        End If
```

اختيار البنود Selecting Items

تسمح أداة صندوق القائمة للمستخدم باختيار إما واحد أو العديد من البنود بالاعتماد على إعدادات خاصية "الصيغة المختارة SelectionMode ". في أداة صندوق القائمة ذات الاختيار الوحيد تستطيع استخلاص البند المختار باستخدام الخاصية"البند المختار SelectedItem " وفهرسها باستخدام الخاصية"الفهرس المختار SelectedIndex".تعود الخاصية "البند المختار SelectedItem " بالبند المختار والذي هو كائن.نص البند المختار يتم التبليغ عنه بواسطة خاصية النصText
إذا كانت الأداة تسمح باختيار العديد من البنود, يتم الإبلاغ reported(تثبيت)باستخدام الخاصية"البنود المختارة SelectedItems "وهذه الخاصية هي جمع من الكائنات وتعرض نفس المكونات التي في جمع البنود.ولأن أداة الصندوق المركب لا تسمح باختيار العديد من البنود, فإنها تزود فقط بخاصيات " SelectedItem البند المختار" و"الفهرس المختار SelectedIndex ". من أجل الدوران على جميع البنود المختارة في أداة صندوق القائمة المتعددة الاختيار,استخدم حلقة مثل التالي:

```
Dim itm As Object
        For Each itm In ListBox1.SelectedItems
            Debug.WriteLine(itm)
        Next
```

المتغير itm يجب أن يتم الإعلان عنه ككائن لان البنود في صندوق القائمة هي كائنات.فإذا كانت جميعها من نفس النوع,تستطيع ان تحولها إلى نوع معين ومن ثم تستدعي طرقها.فإذا كانت جميع البنود من نوع المربع مثلا،تستطيع أن تستخدم حلقة كالتالية لطباعة مساحة كل مربع:

```
Dim itm As Rectangle
        For Each itm In ListBox1.SelectedItems
            Debug.WriteLine(itm.Width * itm.Height)
        Next
```

تدريب مشروع توضيح صندوق القائمة

يوضح هذا المشروع العمليات الأساسية لأداة صندوق القائمة.حيث ان كلا أداتي صندوق القائمة اللتان على الفورم تعملان بشكل مختلف نوعا ما,فالأداة الأولى لديها التركيب الافتراضي:يمكن اختيار بند واحد في نفس الوقت،والبند الجديد يتم إلحاقه بعد البند الموجود.أما أداة صندوق القائمة الثانية فان خاصية "Sorted"الترتيب "تم إعدادها إلى True صح وخاصية "تعدد الاختيار MultiSelect " تم إعدادها تبعا للقيمة في أداتي أزرار التبديل RadioButtonأسفل الفورم.

إن كود تطبيق"توضيح صندوق القائمة" يحتوي الكثير من المنطق الذي تحتاجه في إجراءات معالجة صندوق القائمة. إنه يوضح لك كيف تعمل التالي:

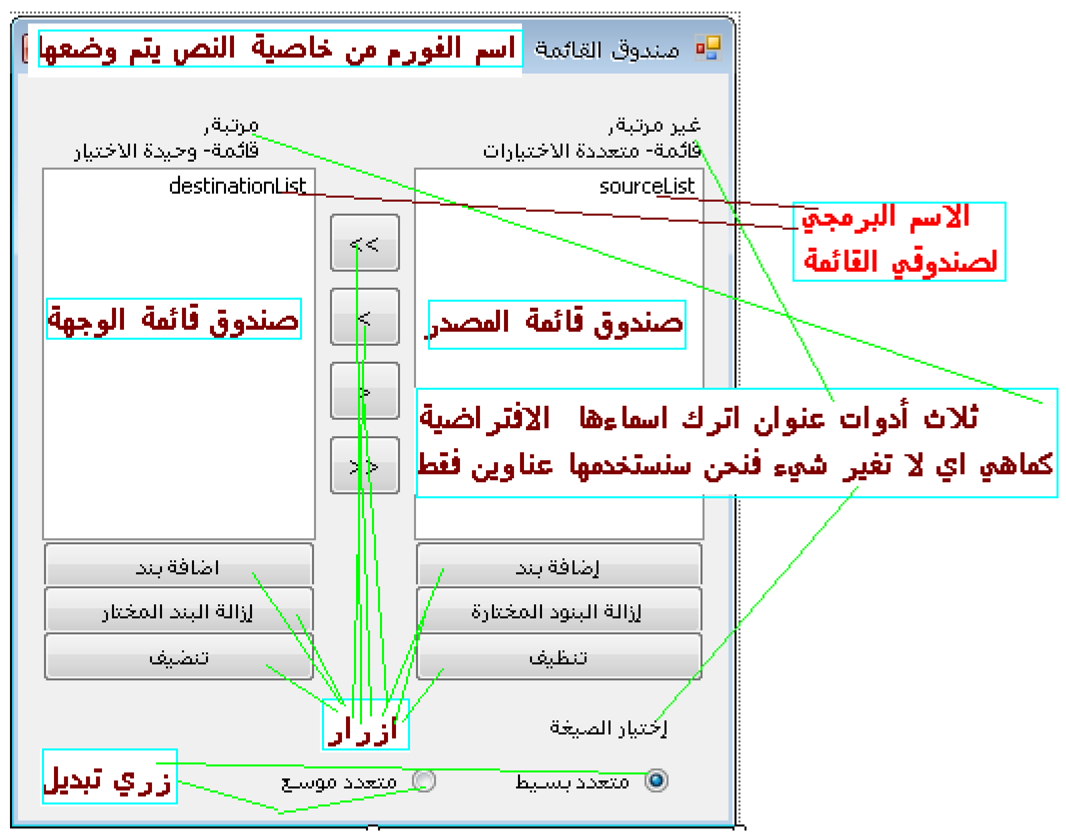
إضافة وإزالة بنود وقت التنفيذ

تحويل البنود بين القوائم وقت التنفيذ

معالجة بنود الاختيار المتعدد

حفظ القوائم المرتبة.

ابدأ مشروع جديد وسمه"مشروع توضيح صندوق القائمة" وعندما يفتح المشروع أعد تسمية الفورم من خلال خاصية النص إلى "صندوق القائمة" وكما هو مبين في الشكل التالي:



الأدوات الموضوعة على الفورم وخصائصها :

اسم الأداة (او اسم خاصية النص text)	الاسم البرمجي للأداة المستخدم في الكود (خاصية name)
ثلاث أدوات عنوانLabel:غيّر خاصية النص text كما مبينة في الشكل	(Label3,Label2,Label1)
صندوق قائمةListBox (لا يوجد لها خاصية نص text)	الاسم البرمجي مبين بالشكل
صندوق قائمة الوجهة ضع لها خاصية (sorted=true)	
أزرار التبديل أسفل الفورم	
زر تبديل RadioButton (متعدد بسيط =text)	radiobtnmultisimple
زر تبديل RadioButton (متعدد موسع=text)	radiobtnmultiextended
الأزرار بين صندوقي القائمة	
زرButton (<< = text)	btnsourcemoveall
زرButton (< = text)	btnsourcemove
زر Button (>= text)	btndestinationmove
زرButton (>> = text)	btndestinationmoveall
الأزرار تحت صندوق القائمة اليسارية	
زرButton (إضافة بند= text)	btndesadd
زرButton (إزالة البند المختار= text)	btndestinationremoves
زرButton (تنظيف= text)	btndestinationclear
الأزرار تحت صندوق القائمة التي على اليمين	
زرButton (إضافة بند= text)	btnsourceadd
زرButton (إزالة البنود المختارة= text)	btnsourceremove
زرButton (تنظيف= text)	btnsourceclear

هذه كل الادوات الخاصيات التي تحتاجها فقط بقي عليك ان تجعل خاصيات من اليمين الى اليسار صحيحة بالنسبة للفورم وكما ذكرنا سابقا من اجل الواجهة الكتابة العربية.الآن إليك الكود كاملا:

```
Public Class Form1
    ' يوضح هذا المشروع كيفية معالجة محتوى أداة صندوق القائمة وقت التنفيذ
    ' 1. كيفية اضافة بنود وقت التنفيذ
    '2. كيفية إزالة البنود المختارة وقت التنفيذ
    '3. كيفية نقل البنود بين القوائم
```

'4. كيفية تبديل صيغة الاختيار لاداة صندوق القائمة

```
    Private Sub btndesadd_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btndesadd.Click
        Dim ListItem As String
        ListItem = InputBox("Enter new item's name")
        If ListItem.Trim <> "" Then
            destinationList.Items.Add(ListItem)
        End If
    End Sub
    Private Sub btnsourceadd_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnsourceadd.Click
        Dim ListItem As String
        ListItem = InputBox("Enter new item's name")
        If ListItem.Trim <> "" Then
            sourceList.Items.Add(ListItem)
        End If
    End Sub
    Private Sub btndestinationremoves_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btndestinationremoves.Click
        destinationList.Items.Remove(destinationList.SelectedItem)
    End Sub
    Private Sub btnsourceremove_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnsourceremove.Click
        Dim i As Integer
        For i = 0 To sourceList.SelectedIndices.Count - 1
' إزالة البند المختار الاول دائما
' لا تستخدم الفهرس (i)لاختيار البند الذي ستتم ازالته
': العبارة التالية
            ' sourceList.Items.RemoveAt(sourceList.SelectedIndices(i))
': لن تزيل البند الصحيح,لان المجمع
' SELECTEDINDICES يتغير في كل مرة تزيل فيها بند
' هذه الحلقة تستمر ف ازالة البنود المختارة في الاعلى
' طالما انها البنود التي تم اختيارها
            sourceList.Items.RemoveAt(sourceList.SelectedIndices(0))
        Next
    End Sub
    Private Sub btndestinationmove_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btndestinationmove.Click
        sourceList.Items.Add(destinationList.SelectedItem)
        destinationList.Items.RemoveAt(destinationList.SelectedIndex)
    End Sub
    Private Sub btnsourcemoveall_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnsourcemoveall.Click
        While sourceList.Items.Count > 0
' حفظ اضافة البند الاعلى الى قائمة الوجهة
' في قائمة المصدر
' هذه الحلقة مشابهة للتي استخدمناها لازالة
' بنود مختارة متعددة من قائمة المصدر_
            destinationList.Items.Add(sourceList.Items(0))
            sourceList.Items.RemoveAt(0)
        End While
    End Sub
    Private Sub btnsourcemove_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnsourcemove.Click
        While sourceList.SelectedIndices.Count > 0
' حفظ اضافة البند المختارالاعلى الى قائمة الوجهة
' في قائمة المصدر
' هذه الحلقة مشابهة للتي استخدمناها لازالة
' بنود مختارة متعددة من قائمة المصدر_
            destinationList.Items.Add(sourceList.Items(sourceList.SelectedIndices(0)))
            sourceList.Items.Remove(sourceList.Items(sourceList.SelectedIndices(0)))
        End While
    End Sub
    Private Sub btnsourceclear_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnsourceclear.Click
        sourceList.Items.Clear()
    End Sub
    Private Sub btndestinationclear_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btndestinationclear.Click
        destinationList.Items.Clear()
    End Sub
    Private Sub btndestinationmoveall_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btndestinationmoveall.Click
        While destinationList.Items.Count > 0
            sourceList.Items.Add(destinationList.Items(0))
            destinationList.Items.RemoveAt(0)
        End While
    End Sub
    Private Sub radiobtnmultisimple_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles radiobtnmultisimple.CheckedChanged, radiobtnmultiextended.CheckedChanged
' وضع خاصية صيغة الاختيار لاداة صندوق قائمة السورس اوالمصدر
' تبعا لاختيار ازرار التبديل
        If radiobtnmultisimple.Checked Then
            sourceList.SelectionMode = SelectionMode.MultiSimple
        Else
```

```
            sourceList.SelectionMode = SelectionMode.MultiExtended
        End If
    End Sub
End Class
```

الكود بسيط ومشروح ضمن الكود من خلال التعليقات ولا أظن أنه بحاجة إلى شرح أكثر.

بحث صندوق القائمة Searching the ListBox

طريقتين من أكثر الطرق فائدة لأداة صندوق القائمة وهما الطريقة FindString والطريقة FindStringExact واللتان تسمحان لك وبسرعة من إيجاد أي بند في القائمة. الطريقة FindString تعمل على إيجاد النص الذي يطابق بشكل جزئي الذي تبحث عنه، أما الطريقة FindStringExact "أوجد النص بالضبط" فإنها تعمل على إيجاد النص المطابق تماما. إذا كنت تبحث عن Man مثلا والأداة تحوي على اسم مثل ***Mansfield*** فان الطريقة "أوجد النص" توافق البند ولكن الطريقة "أوجد النص بالضبط" لا تجد ولا تطابق البند. كل من الطريقتين "أوجد النص" والطريق "أوجد النص بالضبط" تنجزان البحث تحت حالة عدم الحساسية للأحرف case-insensitive. أما الشكل العام لهما فهو التالي:

```
itemIndex = ListBox1.FindString(searchStr As String)
```

حيث أن المتغير searchStr هو النص الذي تبحث عنه. النموذج البديل لكلا الطريقتين يسمح لك بتحديد ترتيب البنود التي سيبدأ عندها البحث:

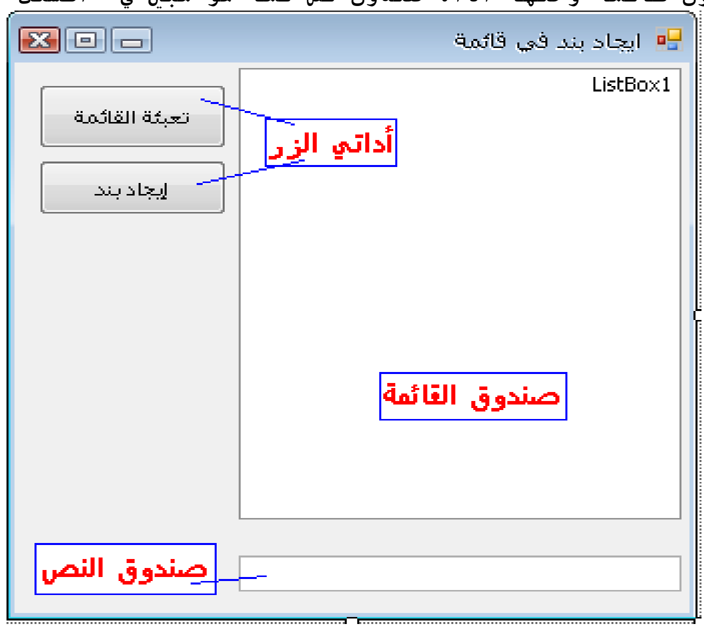
```
itemIndex = ListBox1.FindString(searchStr As String,startIndex As Integer)
```

المعامل النسبي startIndex يسمح لك بتحديد بداية عملية البحث، ولكن لا تستطيع أن تحدد أين سينتهي البحث. الطريقة FindString والطريقة FindStringExact تعملان حتى ولو كانت أداة صندوق القائمة غير مرتبة. فلا تحتاج إلى إعداد الخاصية Sorted إلى صح True قبل أن تستدعي إحدى طرق البحث هذه, ولكن ترتيب القائمة سيساعد في عملية البحث,ولكن تستغرق الأداة أقل من 100 ميلي ثانية لإيجاد بند في قائمة تحوي 100,000 بند لذا فانه لا يستحق صرف الوقت لترتيب القائمة.قبل أن تحمّل الآلاف من البنود في أداة صندوق القائمة ومهما يكن يجب أن تأخذ بعين الاعتبار أن تكون واجهة المستخدم أكثر فعالية.

تدريب: "مشروع أيجاد بند في صندوق القائمة ListBoxFind Application "

التطبيق الذي ستبنيه في هذا المقطع يعمل على تعبئة أداة صندوق القائمة بعدد كبير من البنود ومن ثم إيجاد أي نص تحدده. فالضغط على زر تعبئة يعمل على تعبئة صندوق القائمة بـــ10,000 نص عشوائي.وهذه العملية ستأخذ عدة ثواني وستملأ الأداة بـــنصوص مختلفة وعشوائية كل مرة.من ثم تستطيع إدخال نص في أداة صندوق النص التي في أسفل الفورم.عند ما تكتب الحروف(أو حتى تحذف حروف في صندوق النص) فإن البرنامج سيجد النص المطابق الأقرب في القائمة ويختار(يبرز) هذا البند.

إبدأ مشروع جديد وسمه"مشروع إيجاد بند في قائمة" وسم الفورم الرئيسية له "إيجاد بند في قائمة" ,وكما ذكرنا في ما سبق أجعل الخاصيات من اليمين لليسار لهذه الفورم فعالة.ضع على الفورم زر وسمه من خلال خاصية النص "تعبئة القائمة" واسمه البرمجي"bttnPopulate" وضع زر آخر تحته وسمه " إيجاد بند" واسمه البرمجي"bttnFind" ,وضع على الفورم أيضا أداة صندوق قائمة وتحتها أداة صندوق نص كما هو مبين في الشكل التالي:



يتفاعل هذا التطبيق البسيط مع كل كبسة مفتاح في أداة صندوق النص ويعمل على إيجاد النص الذي تبحث عنه في الحال.زر ايجا بند يعمل نفس الشيء, ولكن أعتقد أنه من الواجب علي توضيح فعالية أداة صندوق القائمة ونوع التخصص الوظيفي الذي ستتوقعه أثناء عمل تطبيقات للزبائن.والآن كود المشروع كاملاً:

```
Public Class Form1
    Private Sub bttnPopulate_Click(ByVal sender As System.Object, ByVal e As    System.EventArgs) Handles
bttnPopulate.Click
        Dim wordLen As Integer
        Dim Nwords As Integer = 99999    ' change this value to generate a longer or shorter list غير هذه القيمة لإنتاج قائمة أطول أو أقصر
        Dim rnd As System.Random
        rnd = New System.Random
        Dim rndChar As Char
        Dim thisWord As String
        Dim i, j As Integer
        ' Disable redrawing while loading itemsإطفاء إعادة السحب أثناء تحميل البنود'
        ListBox1.BeginUpdate()
        ' populate the ListBox with Nwords random wordsتعبئة صندوق القائمة ب س كلمة عشوائية'
        For i = 0 To Nwords
            ' select a random length for the word (from 1 to 25 characters) اختيار طول عشوائي للكلمة(من 1 إلى 25 حرف)
            wordLen = CInt(rnd.NextDouble * 20 + 1)
```

```
                thisWord = ""
                For j = 0 To wordLen
                    ' and build the word by appending random characters
                    ' to the thisWord variable’وبناء الكلمة بإلحاق حروف عشوائية للمتغير"هذه الكلمة"
                    rndChar = Chr(65 + CInt(rnd.Next(0, 25)))
                    thisWord = thisWord & rndChar
                Next
                ' add word to control's Items collection’إضافة كلمة لجمع بنود الأداة
                ListBox1.Items.Add(thisWord)
            Next
            ' we're done, refresh the ListBox control’بعد الانتهاء, تحديث أداة صندوق القائمة
            ListBox1.EndUpdate()
        End Sub
     Private Sub bttnFind_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles bttnFind.Click
            Dim srchWord As String
            Dim wordIndex As Integer
            srchWord = InputBox("ادخل كلمة للبحث عنها")
            ' Attempt to find an exact match with the FindExact method’محاولة إيجاد المطابق تماما بواسطة الطريقة"أوجد بالضبط"
            wordIndex = ListBox1.FindStringExact(srchWord)
            ' If an exact match was found, display the word's index in the Items collection’إذا تم إيجاد المطابق بالضبط,أعرض فهرس الكلمة في مجمع البنود
            If wordIndex >= 0 Then
                ListBox1.TopIndex = wordIndex
                ListBox1.SelectedIndex = wordIndex
                MsgBox("الفهرس = " & wordIndex.ToString & vbCrLf & _
                       "تطابق = " & ListBox1.Items(wordIndex).ToString, , "المطابقةبالضبط")
            Else
                ' Otherwise go for a near match with the FindString method’وإلا إذهب إلى المطابقة القريبة,بالطريقة"إيجاد نص"
                wordIndex = ListBox1.FindString(srchWord)
                If wordIndex >= 0 Then
                    ListBox1.TopIndex = wordIndex
                    ListBox1.SelectedIndex = wordIndex
                    MsgBox("الفهرس = " & wordIndex.ToString & vbCrLf & _
                           "تطابق = " & ListBox1.Items(wordIndex).ToString, , "مطابقة تقريبية")
                Else
                    ' we should never reach this statement, because there will always
                    ' be a near match, unless the list is empty! ’يجب أن لا نصل إلى هذه الحالة(العبارة),لأنه سيكون دائما هناك مطابقة قريبة,غير ذلك فإن القائمة فارغة!
                    MsgBox("البند " & srchWord & " غير موجود في القائمة")
                End If
            End If
        End Sub
     Private Sub TextBox1_TextChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles TextBox1.TextChanged
            Dim srchWord As String = TextBox1.Text.Trim
            If srchWord.Length = 0 Then Exit Sub
            Dim wordIndex As Integer
            ' Attempt to find an exact match with the FindExact method’حاول إيجاد مطابقةبالضبط بواسطة الطريقة"إيجاد المطابق"
            wordIndex = ListBox1.FindStringExact(srchWord)
            ' If an exact match was found, display the word's index in the Items collection’إذا تم إيجاد المطابقة بالضبط,أعرض فهرس الكلمة في مجمع البنود
            If wordIndex >= 0 Then
                ListBox1.TopIndex = wordIndex
                ListBox1.SelectedIndex = wordIndex
                Debug.WriteLine("الفهرس عند المطابقة بالضبط = " & wordIndex.ToString & vbCrLf & _
                       "تطابق = " & ListBox1.Items(wordIndex).ToString)
            Else
                ' Otherwise go for a near match with the FindString method’وإلا إذهب إلى المطابقة القريبة,بواسطة الطريقة"إيجاد نص"
                wordIndex = ListBox1.FindString(srchWord)
                If wordIndex >= 0 Then
                    ListBox1.TopIndex = wordIndex
                    ListBox1.SelectedIndex = wordIndex
                    Debug.WriteLine("الفهرس عند المطابقة التقريبية = " & wordIndex.ToString & vbCrLf & _
                           "تطابق = " & ListBox1.Items(wordIndex).ToString)
                Else
                    ' we should never reach this statement, because there will always
                    ' be a near match, unless the list is empty! ’يجب أن لا نصل إلى هذه الحالة(العبارة),لأنه سيكون دائما هناك مطابقة قريبة,غير ذلك فإن القائمة فارغة!
                    Debug.WriteLine("البند " & srchWord & " غير موجود في القائمة")
                End If
            End If
        End Sub
End Class
```

الكود التالي (ارجع إلى كود "تغير نص صندوق النص" الذي بدايته: **Private Sub** TextBox1_TextChanged). يحاول إيجاد التطابق بالضبط بواسطة الطريقة "إيجاد نص بالضبط FindStringExact " فإذا نجحت هذه العملية,فإنها ترد بفهرس العنصر المطابق.وإلا فإنها تحاول إيجاد مطابقة تقريبية بالطريقة "إيجاد نص FindString " فإذا نجحت فإنها ترد بفهرس المطابقة

التقريبية(والذي هو البند الأول على الأداة الذي يطابق بشكل جزئي معامل البحث النسي)ومن ثم فإنها تنتهي.
في حال الفشل في إيجاد المطابق ,فان البرنامج يظهر رسالة "البند غير موجود في القائمة

تعبئة القائمة Populating the List

زر "تعبئة القائمة" يعمل على إنشاء 10000 بند عشوائي بمساعدة الفئة"عشوائي Random ".في البداية تعمل على إنتاج قيمة عشوائية في المجال من 1 إلى20,والتي هي طول النص(ليس لكل النصوص نفس الطول).ومن ثم فان البرنامج يعمل على إنتاج العديد من الحروف العشوائية بطول النص ويبني النص بإلحاق كل حرف له.هذه الأعداد العشوائية هي في المجال من 65 إلى 91 وهي قيم ألأنسي ANSIللحروف في الحالة الكبيرة.

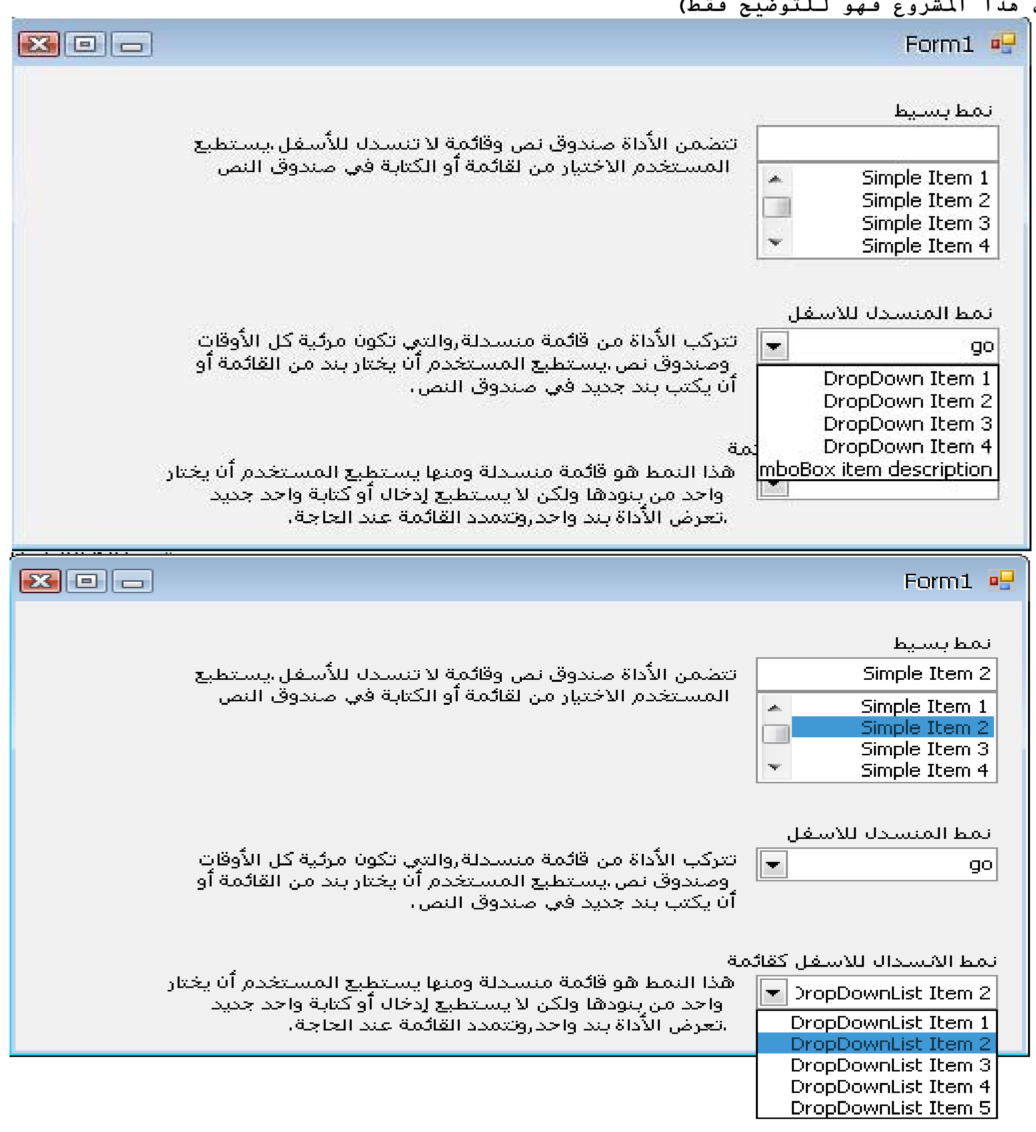
أداة الصندوق المركب The ComboBox Control

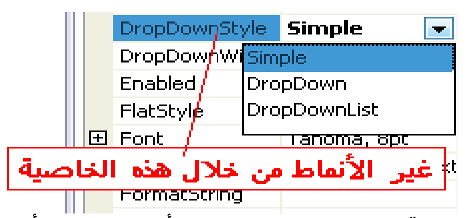
تشابه أداة الصندوق المركب أداة صندوق القائمة في مفهوم أنها تحتوي على بنود عديدة ويمكن للمستخدم أن يختار أحدها,ولكن نموذجيا مساحة أقل على الشاشة.تقريبا الصندوق المركب أداة صندوق قائمة قابلة للتمدد ,حيث يمكن ان يتمدد عندما يريد المستخدم أن يختار ويعيد التقلص بعد انتهاء عملية الاختيار .تعرض أداة الصندوق المركب عادةً سطر مفرد بالبند الذي تم اختياره,وكما لا تسمح هذه الأداة بالاختيارات المتعددة للبنود.فالفرق الأساسي بين اداة الصندوق المركب وأداة صندوق القائمة هو أن صندوق القائمة يسمح للمستخدم بتحديد بنود غير موجودة في القائمة
يوجد ثلاث أنواع من أداة الصندوق المركب .تحدد قيمة الخاصية"نمط Style "لأداة الصندوق المركب أي صندوق تم استخدامه.وهذه القيم مبينة في القائمة التالية:

تبين القائمة التالية "تخطيطات Styles" أداة صندوق القائمة:

القيمة Value	التأثير Effect
DropDown (Default) الانسدال للأسفل(الافتراضية)	تتركب الأداة من قائمة منسدلة,والتي تكون مرئية كل الأوقات وصندوق نص.يستطيع المستخدم أن يختار بند من القائمة أو أن يكتب بند جديد في صندوق النص.
DropDownList الانسدال للأسفل كقائمة	هذا النمط هو قائمة منسدلة ومنها يستطيع المستخدم أن يختار واحد من بنودها ولكن لا يستطيع إدخال أو كتابة واحد جديد .تعرض الأداة بند واحد,وتتمدد القائمة عند الحاجة.
Simple البسيط	تتضمن الأداة صندوق نص وقائمة لا تنسدل للأسفل.يستطيع المستخدم الاختيار من لقائمة أو الكتابة في صندوق النص

يبين الشكلين التالين "مشروع أنماط صندوق القائمة"لتوضيح الأنماط الثلاث لصندوق القائمة والذي هو عنصر مشترك لواجهة النوافذ.(لا تعمل هذا المشروع فهو للتوضيح فقط)





معظم خاصيات وطرق أداة صندوق القائمة ListBox تطبق أيضا على أداة الصندوق المركبComboBox. يمنحك مجمع البنود Items collection إمكانية الوصول إلى بنود الأداة,و يمنحك المجمع "الفهارس المختارة "SelectedIndices والمجمع"البنود المختارة SelectedItems " إمكانية الوصول إلى بنود الاختيار الحالي.إذا كانت الأداة تسمح باختيار ببند وحيد فقط,استخدم الخاصية"الفهرس المختار SelectedIndex " والخاصية"البند المختارselectedItem". تستطيع أيضا أن تستخدم الطريقة"إيجاد نص FindString " والطريقة"إيجاد النص بالضبط "FindStringExact لإيجاد أي بند في الأداة.

توجد سمة تستحق الذكر بخصوص العمليات على الأداة.على الرغم من أن صندوق التحرير في الأعلى(صندوق نص الأداة الذي تستطيع أن تكتب فيه إذا كان ذلك متاح أو يظهر فيه البند الذي تختاره من قائمة الأداة, انظر الأشكال السابقة ولاحظ ظهور البنود التي تختارها أو تكتبه في هذا الصندوق مثل الصندوق المكتوب فيه كلمة go:) يتيح لك إدخال نص جديد فإن النص الجديد لن يصبح بند جديد في القائمة ويبقى هناك حتى تختار بند آخر أو تنظف صندوق التحرير،تستطيع إضافة بعض الكود لجعل أي نص مدخل بواسطة المستخدم في صندوق تحرير الأداة لأن يتم إضافته إلى قائمة البنود الموجودة.

الاستخدام الأكثر شيوعا لأداة الصندوق المركب كجدول بحث(كما هو مبين في الشكل التالي حيث أن القاموس الشهير بابليون يستخدم الصندوق المركب كـأداة بحث وهي ميزة شائعة الاستخدام في القواميس الالكترونية)حيث أن يشغل مساحة قليلة جدا على الفورم,ولكن بإمكانه التمدد متى تشاءwill at. تستطيع حتى توفير saveمساحةعندما يتم تقليص الصندوق المركب بوضعه إلى أقل عرض يناسب البند الأطول.استخدام الخاصية"عرض الانسدال DropDownWidth" والتي هي عرض مقطع قائمة المنسدلة.بشكل افتراضي هذه الخاصية مساوية إلى خاصية عرض الأداة.فصندوق القائمة الثاني في الأشكال السابقة يحتوي على بند طويل واتساع الأداة كاف لعرض الاختيار الافتراضي.عندما ينقر المستخدم على السهم الصغير لتمديد الأداة فإن مقطع الانسدال للأداة أعرض من العرض الافتراضي لذا فإن البند الطويل يمكن قراءته.



إضافة بنود للصندوق المركب وقت التنفيذ Adding Items to a ComboBox at Runtime

على الرغم من أن أداة الصندوق المركب تتيح للمستخدم إدخال نص في صندوق تحرير الأداة.ولكنها لا توفر آلية بسيطة لإضافة بند جديد وقت التشغيل,لنقول أنك زودت صندوق مركب بأسماء المدن, يستطيع المستخدم كتابة الأحرف الأولى وبسرعة يتم إيجاد البند المرغوب.ولكن إذا كنت تريد أن تتيح للمستخدمين إضافة أسماء مدن جديدة, تستطيع أن توفر هذه الميزة من خلال تقنيتان بسيطتان.الأبسط منهما هي وضع زر عليه علامة إضمارellipsis (ثلاث نقط) قرب الأداة تماما.عندما يريد المستخدمين إضافة بند جديد للأداة بإمكانهم النقر على هذا الزر ويطلب منهم إدخال البند الجديد.

المقاربة الأكثر لباقة هي تفحص خاصية النص للأداة حالما تفقد الأداة التركيز,أو حالما يضغط المستخدم مفتاح الإدخال.فإذا كان النص المدخل بواسطة المستخدم لا يطابق أي بند موضوع على الأداة, يتوجب عليك إضافة بند جديد إلى مجمع بنود الأداة واختيار البند الجديد من ضمن الكود."مشروع الصندوق المركب المرن" يوضح كيفية استخدام كلا التقنيتان في كودك.الفورم الرئيسية للمشروع والتي تظهر في الشكل التالي هي نافذة إدخال بيانات بسيطة .ولكنها ليست الفورم الأفضل لإدخال البيانات,والقصد منها التوضيح فقط.

تستطيع إما إدخال اسم المدينة(أو اسم البلد)ومن ثم الضغط على مفتاح التنقلTab,للانتقال إلى أداة أخرى أو النقر على الزر جانب الأداة ليطلب منك اسم المدينة/البلد.سيتيح لك التطبيق ضم أي مدينة/بلد.

يجب أن تزود بالكود لحصر المدن بالبلد المختار,ولكن هذه مهمة غير عادية.تحتاج أيضا إلى تخزين أسماء المدن الجديدة المدخلة على أداة الصندوق المركب الأولى في ملف(أو جدول قاعدة بيانات),لذا فإن المستخدمين بإمكانهم إيجادها هناك في المرة القادمة عندما ينفذون التطبيق.لم أجعل التطبيق محكما,فقد عملت فقط على إضافة الكود اللازم لتوضيح كيفية إضافة بنود إلى أداة الصندوق المركب وقت التنفيذ.

تدريب: مشروع الصندوق المركب المرن

ابدأ مشروع جديد وسمه"مشروع الصندوق المركب المرن" وسم الفورم"الصندوق المركب المرن" وضع عليه الأدوات المبينة في الشكل التالي:

الصندوق المركب المرن

الاسم

العنوان

المدينة -- البلد ...

الرقم البريدي

قبول رفض

في هذا المشروع سنعمل على تعديل الخاصيات التالية للأدوات التي على الفورم:
بالنسبة لأدوات العناوين(label)فإننا لم نعمل عليها أي تعديل فقط سنغير خاصية الاسم لتظهر العناوين كما هي مبينة في الشكل السابق و العناوين هي(الاسم و العنوان,و المدينة,و البلد,و الرقم البريدي بالترتيب من الأعلى إلى الأسفل)

الأداة	الخاصية
أدوات صناديق النصوص(textbox)	
صندوق نص الاسم	الاسم البرمجي (name= txtName)
صندوق نص العنوان	الاسم البرمجي (name= txtAddress)
صندوق نص الرقم البريدي	الاسم البرمجي (name= txtPostalCode)
أدوات الصناديق المركبة(combobox)	
صندوق المركب للمدينة	الاسم البرمجي(name= cbCity)
صندوق المركب للبلد	الاسم البرمجي(name= cbCountry)
الأزرار(button)	
زر قبول	الاسم البرمجي(name= bttnOK)
زر رفض	الاسم البرمجي(name= bttnCancel)
زر النقل للمدينة (--)	الاسم البرمجي(name= bttnLocateCity)
زر النقل للبلد (...)	الاسم البرمجي(name= bttnLocateCountry)

زر النقل ellipsis buttonالمجاور لأداة الصندوق المركب"المدينة"يطلب المستخدم من أجل بند جديد بواسطة الوظيفة "صندوق الإدخال InputBox() ".ومن ثم يبحث عن البند في جمع بنود الأداة بواسطة الطريقة "إيجاد نص FindString " وإذا كان البند الجديد غير موجود فإنه يعمل على إضافته إلى الأداة.ومن ثم فإن الكود يختار البند الجديد في القائمة.ليعمل هذا,فإنه يضع خاصية الأداة"الفهرس المختار SelectedIndex " إلى القيمة المعاد بواسطة الطريقة" إضافة.بنود Items.Add) أو القيمة المعادة بواسطة الطريقة"إيجاد نص FindString " بالاعتماد على إما أن البند تم إيجاده أو تم إضافته إلى القائمة.الكود التالي يبين كيفية إضافة بند جديد إلى أداة صندوق المركب وقت التنفيذ:

```
Private Sub Button1 Click(...) Button1.Click
        Dim itm As String
        itm = InputBox("Enter new item", "New Item")
        If itm.Trim <> "" Then AddElement(itm)
    End Sub
```

الإجراء الجزئي"إضافة عنصر AddElement" والذي يقبل نص كمعامل نسي ويضيفه إلى الأداة المبين في الكود(في الكود الكامل(راجع الكود ف الأسفل).فإذا كان البند غير موجود في الأداة يتم إضافته إلى الجمع"بنود Items " أما إذا كان البند عضو في جمع"البنود Items " يتم اختياره.كما هو مبين في كود هذا الإجراء,نفس هذا الأجراء سيتم استخدامه بالطريقة الثانية لإضافة بنود وقت التنفيذ(من اجل إدخال اسم البلد).

والآن إليك الكود كامل:

```
Public Class Form1
     Private Sub Form1_KeyUp(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyEventArgs) Handles
Me.KeyUp
        If e.KeyCode = Keys.Enter And (Me.ActiveControl.GetType Is GetType(TextBox) Or
Me.ActiveControl.GetType Is GetType(ComboBox)) Then
            Me.ProcessTabKey(True)
        End If
    End Sub
     Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        cbCity.SelectedIndex = -1
        cbCountry.SelectedIndex = -1
    End Sub
     Private Sub ClearFields()
        txtAddress.Clear()
        txtName.Clear()
        txtPostalCode.Clear()
        cbCity.SelectedIndex = -1
        cbCountry.SelectedIndex = -1
```

```
    End Sub
     Sub AddElement(ByRef control As ComboBox, ByVal newItem As String)
        Dim idx As Integer
        If control.FindString(newItem) >= 0 Then
            idx = control.FindString(newItem)
        Else
            idx = control.Items.Add(newItem)
        End If
        control.SelectedIndex = idx
    End Sub
     Private Sub cbCity_LostFocus(ByVal sender As Object, ByVal e As System.EventArgs) Handles
cbCity.LostFocus
        Dim newItem As String = cbCity.Text.Trim
        If newItem <> "" Then AddElement(sender, newItem)
    End Sub
     Private Sub cbCountry_LostFocus(ByVal sender As Object, ByVal e As System.EventArgs) Handles
cbCountry.LostFocus
        Dim newItem As String = cbCountry.Text.Trim
        If newItem <> "" Then AddElement(sender, newItem)
    End Sub
     Private Sub bttnLocateCity_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
bttnLocateCity.Click
        Dim itm As String
        itm = InputBox("ادخل اسم المدينة", "مدينة جديدة")
        If itm <> "" Then
            AddElement(cbCity, itm)
        End If
    End Sub
    Private Sub bttnLocateCountry_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
bttnLocateCountry.Click
        Dim itm As String
        itm = InputBox("أدخل اسم البلد", "بلد جديد")
        If itm <> "" Then
            AddElement(cbCountry, itm)
        End If
    End Sub
     Private Sub bttnOK_Click(ByVal sender As System.Object, ByVal e As  System.EventArgs) Handles
bttnOK.Click
        Dim msg As String
        msg = "الاسم " & vbTab & vbTab & txtName.Text & vbCrLf
        msg &= "العنوان " & vbTab & vbTab & txtAddress.Text & vbCrLf
        msg &= "المدينة " & vbTab & vbTab & cbCity.Text & vbCrLf
        msg &= "البلد " & vbTab & cbCountry.Text & vbCrLf
        msg &= "الرقم البريدي " & vbTab & vbTab & txtPostalCode.Text & vbCrLf
        MsgBox(msg, MsgBoxStyle.OkOnly, "قدمت البيانات التالية:")
        ClearFields()
        txtName.Focus()
    End Sub
    Private Sub bttnCancel_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
bttnCancel.Click
        ClearFields()
        txtName.Focus()
    End Sub
End Class
```

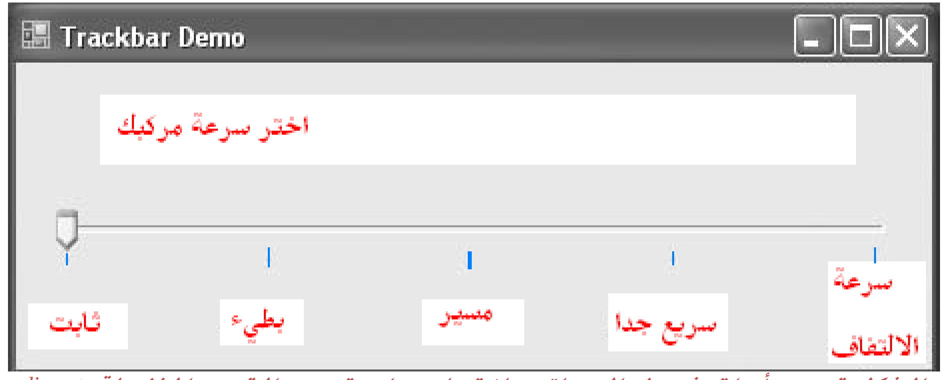
تستطيع أيضا إضافة بنود جديدة وقت التنفيذ بإضافة نفس الكود في معالج حدث الأداة "فقد التركيز LostFocus ":

```
Private Sub ComboBox1 LostFocus(...) Handles ComboBox1.LostFocus
        Dim newItem As String = ComboBox1.Text
        AddElement(newItem)
    End Sub
```

أداة شريط الانزلاق وأداة شريط السياق:The ScrollBar and TrackBar Controls

تسمح لك أداة شريط الانزلاق ScrollBar وأداة شريط السياق TrackBar بتخصيص الحجم, بسحب الناخب **selector** بين القيمتين الأكبر والأصغر.في بعض الحالات لا يعلم المستخدم مقدما قيمة الكمية التي عليه تخصيصها(الحالة التي يفي فيها صندوق النص بالغرض مثلا)لذا فإن تطبيقك يجب أن يوفر آلية أكثر مرونة لتخصيص القيمة,من خلال بعض الأنواع المرئية للتغذية الراجعة feedback .

يسمح شريط الانزلاق الشاقولي للمستخدم بتحريك المستند الطويل للأعلى أو الأسفل وهو مثال نموذجي لاستخدام أداة شريط الانزلاق.شريط الانزلاق والتغذية الراجعة المرئية visual feedback هما الآليتان الرئيسيتان من أجل إعادة تموضع المشهد في المستند الطويل أو في الصورة الكبيرة والتي لا تتناسب بشكل كامل في نافذتها.أداة شريط السياق تماثل أداة شريط الانزلاق,ولكنها لا تغطي مجال مستمر(متواصل) من القيم.حيث أن أداة شريط السياق لها عدد ثابت من علامات التدريج(التقسيم tick marks)حيث يستطيع المبرمج أن يعنون (مثلا ثابت,بطيء,سرعة الدوران(الانعطاف) كما مبين في الشكل).يمكن للمستخدمين من وضع مؤشر المنزلقة إلى القيمة المرغوبة,بينما أداة شريط الانزلاق تعتمد relies على بعض التغذية الراجعة المرئية خارج إطار الأداة لمساعدة المستخدم في وضع المؤشر إلى القيمة المرغوبة,جبر أداة شريط السياق المستخدم على الاختيار من مجال لقيم مشروعة.



الشكل:تسمح أداة شريط السياق باختيار واحدة من القيم المنفصلة discrete

باختصار, سيتم استخدام أداة شريط الانزلاق عندما تكون القيمة بالضبط(القيمة الدقيقة) غير مهمة بقدر أهمية تأثير القيمة على كائن أخر أو عنصر بيانات(أي تأثير القيمة أهم من دقتها).وسيتم استخدام أداة شريط السياق عندما يكون بإمكان المستخدم كتابة قيمة عددية والقيمة التي يتوقعها تطبيقك هي عدد ضمن مجال محدد,مثلا الأعداد الصحيحة بين 0 و100 أو القيمة بين 0 و5إنش بخطوات بطول 0.1إنش(0.0, 0.1, 0.2 . . . 5.0).يتم تفضيل أداة شريط السياق بالنسبة لأداة صندوق النص في الحالات المتشابهة لأنه ليس هناك حاجة للتحقق من البيانات من جهتك.فالمستخدم لا يستطيع تخصيص إلا القيم العددية المتاحة بالفارة.

أداة شريط الانزلاق The ScrollBar Control

لا توجد أداة شريط الانزلاق بحد ذاتها per se في صندوق الأدوات,عوضا عنها يوجد تنوعان لها:أداة شريط الانزلاق الأفقي HScrollBar وأداة شريط الانزلاق الشاقولي VScrollBar.يختلفان فقط في اتجاههما، ولكن ولأنهما يشتركان بنفس المكونات فإني سأشير إليهما مجتمعتان كـأدوات شريط الانزلاق.فعليا كلاهما يرث inherit من أداة شريط الانزلاق,والتي هي أداة مطلقة abstract:يمكن استخدامها لتنفيذ أشرطة انزلاق أفقية و شاقولية ولكن لا يمكن استخدامها مباشرة على الفورم.وأكثر من ذلك فإن أداة الانزلاق الأفقية والشاقولية غير معروضتان في سوية(عروة tab) الأدوات المشتركة Common Controls لصندوق الأدوات Toolbox. عليك فتح سوية(لسان)جميع نماذج ويندوز Windows Forms لتجد هاتان الأداتان.

أداة شريط الانزلاق ScrollBarهي شريط stripطويل مع مؤشر يتيح للمستخدم باختيار قيمة من بين نهايتي الأداة.النهاية اليسارية(السفلى) للأداة تتوافق مع القيمة الدنيا minimum, أما النهاية الأخرى فهي القيمة العظمى maximum للأداة . القيمة الحالية للأداة هي المحددة بواسطة موضع المؤشر,والذي يمكن زلقه بين القيمة الصغرى والعظمى.الخاصية الأساسية لأداة شريط الانزلاق إذا تم تسميتها بشكل مناسب وهي القيمة الصغرى Minimum والعظمى Maximum .

خاصية الصغرى Minimum:قيمة الأداة الصغرى. القيمة الافتراضية هي 0 ولكن وبسبب أنها قيمة صحيحة تستطيع وضعها إلى قيمة سالبة إذا أردت.

خاصية العظمى Maximum: قيمة الأداة العظمى.القيمة الافتراضية 100 ولكن تستطيع وضعها إلى أي قيمة بحيث تستطيع أن تمثلها بنوع بيانات عددية صحيحة Integer data type .

خاصية القيمة Value: القيمة الحالية للأداة, محددة بواسطة موضع المؤشر.

الخاصية Minimumوالخاصية Maximumهي قيم صحيحة.لتغطية مجال غير القيم الصحيحة nonintegers, عليك توفير كود لربط القيم الفعلية بقيم صحيحة. مثلا لتغطية مجال من 2.5 إلى8.5 ضع خاصية الصغرى Minimumإلى 25 وخاصية العظمى Maximumإلى85 وقسم قيمة valueالأداة على عشرة10 . إذا كان المجال الذي تحتاجه من-25 إلى 85,افعل المثل ولكن ضع خاصية Minimumالصغرى إلى-25 وخاصية Maximumالعظمى إلى 85وقسم خاصية Value القيمة على 10 .

يوجد خاصيتان أيضا تسمحان لك بالتحكم بحركة المؤشر:خاصية تغير صغير SmallChangeوخاصية تغير كبير LargeChange.الخاصية الأولى هي الكمية التي يتغير بها المؤشر عندما ينقر المستخدم الأسهم التي عند نهايتي الأداة,خاصية التغير الكبير LargeChange الإزاحة للمؤشر عندما ينقر المستخدم في مكان على شريط الانزلاق نفسه.تستطيع معالجة شريط الانزلاق باستخدام لوحة المفاتيح أيضا,ضغط مفتاح السهم arrowلتحريك المؤشر في الاتجاه الموافق بتغير صغيرSmallChange. والمفاتيح صفحة للأعلىPageUp/PageDown صفحة للأسفل لتحريك المؤشر بتغير كبيرLargeChange .

تدريب:مشروع الألوان The Colors Project

يسمح هذا المشروع البسيط للمستخدم بتخصيص لون ما بمعالجة قيمة الألوان الأساسية (الأحمر,الأخضر،والأزرق) من خلال أشرطة انزلاق.فكل لون أساسي تم التحكم به بواسطة شريط انزلاق ولديه قيمة صغرى هي الصفر وقيمة عظمى وهي 255 . إذا كنت لا تعرف تعريف الألوان في بيئة ويندوز انظر المقطع"تخصيص الألوان Specifying Colors " في الفصل 19 "معالج الصور والمجسماتManipulating Images and Bitmaps "

عند تحريك مؤشر شريط الانزلاق,فإن اللون الموافق يتم عرضه,ويستطيع المستخدم بسهولة تخصيص لون ما بدون معرفة القيم الدقيقة لمكوناته الرئيسية.يحتاج جميع المستخدمين لمعرفة فيماإذا كان اللون المرغوب يحتوي, مثلا على الكثير من الأحمر أو القليل من الأخضر.بمساعدة أشرطة الانزلاق والتغذية الراجعة المباشرة من التطبيق,يمكن للمستخدم بسهولة أن يعين بدقة اللون المطلوب.لاحظ أن القيم الدقيقة للمكونات الأساسية للون ليس لها أهمية عملية،فقط المهم إحصائيات اللون الأخير.

ضع الأدوات المبينة في الشكل على الفورم واعمل على إعداد الخاصيات التالية لها:

الألوان

أداتي صندوق صورة

hscrollbarأدوات أشرطة الانزلاق الافقية

أدوات العنوان

أحمر

أخضر

أزرق

الأداة	الخاصية
صندوق الصورة PictureBox1	borderstyle= Fixed3D
صندوق الصورة PictureBox2	borderstyle= Fixed3D
أدوات العنوان label	
أحمر	text =أحمر Forecolor=red Name= lblRed
أخضر	text =أخضر Forecolor=green Name= lblGreen
أزرق	text =أزرق Forecolor=blue Name= lblBlue
أشرطة الانزلاق HScrollBar	
شريط انزلاق أحمر	name= redBar
شريط انزلاق أحمر	name= greenBar
شريط انزلاق أحمر	name= blueBar

إليك الآن الكود كامل:

```
Private Sub redBar_Scroll(ByVal sender As System.Object, ByVal e As System.Windows.Forms.ScrollEventArgs)
Handles redBar.Scroll
        If e.Type = ScrollEventType.EndScroll Then ColorBox1()
        lblRed.Text = "RED " & redBar.Value.ToString("###")
    End Sub
     Private Sub greenBar_Scroll(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.ScrollEventArgs) Handles greenBar.Scroll
        If e.Type = ScrollEventType.EndScroll Then ColorBox1()
        lblGreen.Text = "GREEN " & greenBar.Value.ToString("###")
    End Sub
    Private Sub blueBar_Scroll(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.ScrollEventArgs) Handles blueBar.Scroll
        If e.Type = ScrollEventType.EndScroll Then ColorBox1()
        lblBlue.Text = "BLUE " & blueBar.Value.ToString("###")
    End Sub
     Private Sub redBar_ValueChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles
redBar.ValueChanged
        ColorBox2()
    End Sub
    Private Sub greenBar_ValueChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles
greenBar.ValueChanged
        ColorBox2()
    End Sub
     Private Sub blueBar_ValueChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles
blueBar.ValueChanged
        ColorBox2()
    End Sub
    Sub ColorBox1()
        Dim clr As Color
        clr = Color.FromArgb(redBar.Value, greenBar.Value, blueBar.Value)
        PictureBox1.BackColor = clr
    End Sub
     Sub ColorBox2()
        Dim clr As Color
        clr = Color.FromArgb(redBar.Value, greenBar.Value, blueBar.Value)
        PictureBox2.BackColor = clr
    End Sub
```

أحداث أداة شريط الانزلاق The ScrollBar Control's Events

يمكن للمستخدم أن يغير قيمة أداة شريط الانزلاق بثلاث سبل:بالضغط على السهمين الذين عند كلا نهايتيها،بضغط المنطقة الموجودة بين المؤشر والأسهم,وبسحب المؤشر بالفارة.بإمكانك عرض التغيرات لقيمة شريط الانزلاق من ضمن كودك باستخدام حدثين:"القيمة تغيرت ValueChanged " وانزلاقScroll .كلا الحدثين يتم إطلاقهما في كل مرة يتغير فيها موضع المؤشر.إذا غيرت قيمة الأداة من ضمن كودك،فإن الحدث"القيمة تغيرت ValueChanged" هو الوحيد الذي سيتم إطلاقه.

الحدث "انزلاق Scroll"يمكن إطلاقه في حالة الاستجابة للعديد من الأفعال،مثل زلق(سحب) المؤشر بالفارة,الضغط على أحد الأزرار الموجودة عند نهايتي الأداة,وهكذا.إذا أردت معرفة الفعل الذي سبب هذا الحدث.تستطيع اختبار الخاصية"نوع Type" للمعامل النسبي الثاني من معالج الحدث.إعدادات الخاصية"نوع الحدث e.Typeي"هي عضو في عداد "نوع حدث الانزلاق ScrollEventType"(تناقص كبير,زيادة صغيرة,سياق,وهكذا)

معالجة الأحداث في تطبيق الألوان

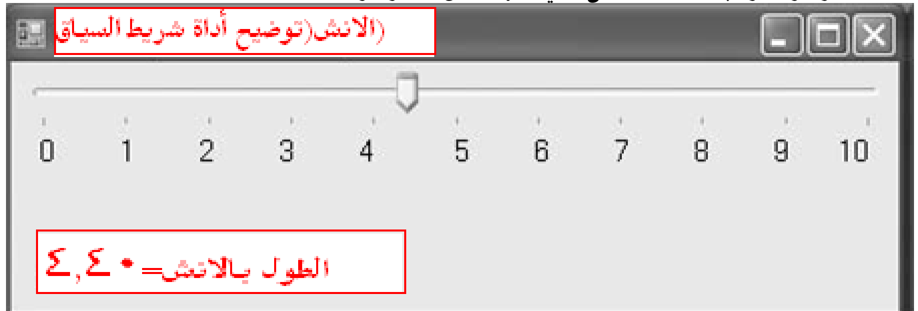
يوضح تطبيق الألوان كيفية برمجة كلا الحدثين لأداة شريط الانزلاق.تعرض أداتي صندوق الصورة اللون المصمم بواسطة أدوات أشرطة الانزلاق الثلاث,صندوق الصورة PictureBox الذي على اليسار يتم تلوينه من ضمن الحدث"انزلاق Scroll",بينما الأخر يتم تلوينه من ضمن الحدث "القيمة تغيرت ValueChanged" تم إطلاق كلا الحدثين عندما يسحب المستخدم مؤشر شريط الانزلاق.ولكن في معالج حدث الانزلاق Scrollلأشرطة الانزلاق الثلاث,يتفحص الكود قيمة الخاصية" e.Type" ويتفاعل معها فقط إذا تم إطلاق الحدث لأن سحب المؤشر قد انتهى.من أجل جميع الأفعال الأخرى لا يعمل معالج الحدث على تحديث اللون لصندوق الصورة الذي على اليسار.إذا حاول المستخدم تغير قيمة اللون بالضغط على كلا السهمين لشريط الانزلاق أو بالضغط في المنطقة على يسار أو يمين المؤشر.لون صندوقي الصورة يتم تحديثه.بينما يزلق المستخدم المؤشر أو يبقى ضاغطاً على واحد من أسهم أدوات شريط الانزلاق فإن الصورة التي على اليمين هي فقط التي يتم تحديثها.

النتيجة من هذه التجربة هو أنك تستطيع برمجة إما حدث لتوفير تغذية راجعة feedbackمستمرة للمستخدم.إذا كانت هذه التغذية الراجعة تتطلب الكثير من الحسابات,والتي ستبطئ استجابة معالج الحدث الموافق.تستطيع تأجيل postpone الاستجابة حتى يتوقف المستخدم عن سحب المؤشر.تستطيع التقاط detectهذه الحالة باختبار القيمة للخاصيةe.Type.عند القيمة "نوع حدث الزلق.انتهى الزلق: ScrollEventType.EndScroll",تستطيع تنفيذ العبارات المناسبة.(راجع الكود خلف شريط الانزلاق للون الأحمر للحدث "زلق Scroll" والحدث"القيمة تغيرت ValueChanged".كود حدث الاستجابة بالنسبة لأدوات أشرطة الانزلاق الأخرى مشابه.

يعمل الإجراء () ColorBox1وColorBox2على تحديث لون أداتي صندوق الصورة بوضع لون الخلفية لكل منهما.

أداة شريط السياق The TrackBar Control

إن أداة شريط السياق مشابهة تماما لأداة شريط الانزلاق,ولكن أداة شريط الانزلاق تفتقر lacks إلى التدريجات (التقسيمات granularity).على فرض أنك أردت من مستخدم التطبيق أن يعطي قيمة ضمن مجال معين,مثل سرعة كائن (جسم)متحرك,وأكثر من ذلك,فأنت لا تريد السماح بالزيادة المفرطة في الدقةprecision.ما تحتاجه فقط عدة إعدادات,كما هو مبين في الشكل السابق(اختر سرعة المركب)والشكل التالي,يستطيع المستخدم أن يضع قيمة الأداة بزحلقة slidingالمؤشر أو بالضغط على أي جهة من المؤشر.



الحبيبية(التدريجات Granularity)هي بأي تخصيص(تحديد) تريد أن يكون القياس .في قياس المسافات بين المدن تكون تدريجات الميل كافية quite adequateتماما(ملائمة).في قياس(تحديد) الأبعاد لبناء ما يمكن للتدريجات أن تكون بقياس القدم,أو الانش,أداة شريط السياق تسمح لك بوضع نوع المقياس الضروري لتطبيقك.بشكل مشابه لأداة شريط الانزلاق,الخاصية SmallChange والخاصية LargeChangeمتاحتان,و"التغير البسيط SmallChange"هي الزيادة الأقل التي يمكن أن تتغير بها قيمة المنزلقة Slider.يستطيع المستخدم أن يغير هذه المنزلقة بواسطة قيمة"التغير البسيط" بزلق المؤشر فقط .(لا تشبه أداة شريط الانزلاق فلا يوجد سهمين عند نهايتي أداة المنزلقة Slider) لتغير قيمة المنزلقة بالخاصية"تغير كبيرLargeChange"يستطيع المستخدم أن يضغط على أي من جهتي المؤشر.

تدريب:مشروع الإنش The Inches Project

يوضح الشكل السابق استخدام نموذجي لأداة شريط السياق.الفورم في الشكل السابق هي عنصر واجهة مستخدم البرنامج والتي تسمح للمستخدم بتخصيص المسافة بين الصفر0 والعشرة10 بالإنش,وبزيادة بمقدار0.2انش.عندما يزحلق المستخدم المؤشر,فإنه يتم عرض القيمة الحالية على أداة عنوان أسفل شريط السياق.

يمكن أن تحدد للأداة 50 توقف(تقسيمات)ولكن فقط 10 منهم ستكون مرئية.يستطيع المستخدم ومهما يكن وضع المؤشر على أي من علامات الوقف(التيك) الـ40 الغير مرئية.يمكن أن تفكر في أن العلامات المرئية كعلامات تدريج رئيسية major,والعلامات غير المرئية كعلامات ثانوية minor.إذا كانت خاصية " TickFrequency" هي 5,سيكون مرئي فقط 5 علامات,وسيتوقف مؤشر المنزلقة عند جميع علامات الوقف(التأشير) أو التدريجtick marks.

عندما تستخدم أداة شريط السياق على واجهة تطبيقك,عليك إعداد خاصية" TickFrequency" إلى قيمة تساعد المستخدم باختيار الأعداد المطلوب,الكثير من علامات الوقف تكون مربكة وصعبة القراءة.وبدون علامات الوقف,لا تساعد الأداة كثيرا,من الممكن أيضا أن تضع القليل من أدوات العنوان لتشير إلى قيمة العلامات المختارة.كما فعلت في هذا المثال.

خاصيات أداة شريط السياق في تطبيق "الإنش" هي التالية:

خاصية الأصغر Minimum=0

خاصية الأعظم Maximum=50

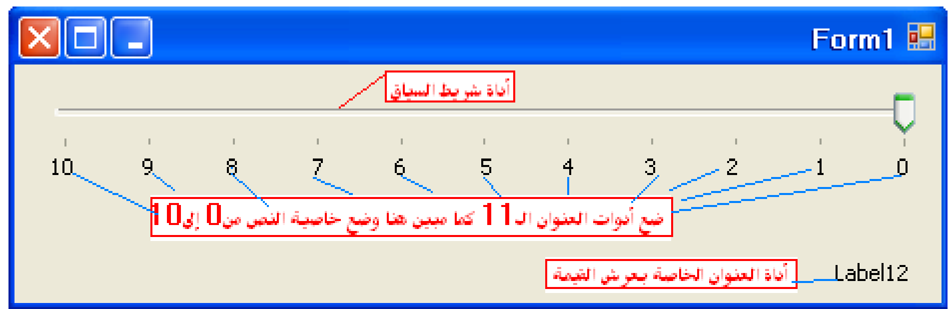
خاصية"التغير البسيط SmallChange"=1

خاصية "التغير الكبير LargeChange"=5

خاصية"تتابع الوقفTickFrequency"=5

تحتاج أداة شريط السياق أن تغطي المجال 10انش بزيادة قدرها0.2انش.إذا وضعت خاصية"التغير البسيط"إلى1 عليك وضع خاصية "التغير لكبير"إلى5,وأكثر من ذلك تم وضع خاصية"تكرار الوقف"إلى 5,لذا سيكون مجموع من خمس تقسيمات في كل إنش.الأعداد تحت علامات الوقف تم وضعها هناك بواسطة أدوات عنوان مرصوفة.وأداة العنوان التي في الأسفل تحتاج إلى تحديث عند تغير قيمة شريط السياق.تمت الإشارة إليها في التطبيق بحدث"التغير Change",والذي يحدث في كل

مرة تتغير فيها قيمة الأداة,سواء من خلال السحب أو من ضمن الكود.(راجع كود **ValueChanged event handler** معالج حدث تغير القيمة لأداة شريط السياق **TrackBar**)
ولآن افتح مشروع جديد وسمه "مشروع الإنش" وضع عليه الأدوات المبينة في الشكل وضع خاصيات من اليمين إلى اليسار ولا تنسى وضع الخاصيات المذكورة في الأعلى بالنسبة لأداة شريط السياق ,نسق هذا الفورم بحيث يبدو كما هو مبين في الشكل التالي:



والآن إليك الكود كامل للمشروع(جربه وحرك مؤشر الانزلاق بالفارة ومن ثم حركه بواسطة الأسهم التي على لوحة المفاتيح ولاحظ كيف تظهر قيمة المنزلقة في أداة العنوان التي في الأسفل والخاصة بعرض القيمة Label12)

```
Public Class Form1
    Private Sub Label_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) _
                      Handles Label1.Click, Label2.Click, _
                      Label3.Click, Label4.Click, Label5.Click, Label6.Click, _
                      Label7.Click, Label8.Click, Label9.Click, Label10.Click, Label11.Click
        TrackBar1.Value = CInt(CType(sender, Label).Text) * 5
    End Sub
    Private Sub TrackBar1_ValueChanged(ByVal sender As Object, ByVal e As System.EventArgs)
Handles TrackBar1.ValueChanged
        Label12.Text = "الطول بالانش = " & _
              Format(TrackBar1.Value / 5, "#.00")
    End Sub
End Class
```

أدوات العنوان Label التي أسفل علامات الوقف يمكن أن تستخدم لوضع قيمة للأداة في كل مرة يتم النقر على واحدة من هذه الأدوات,(شاهد كود معالج نقر أدوات العنوان في الأعلى والذي يبدأ Private Sub Label_Click).